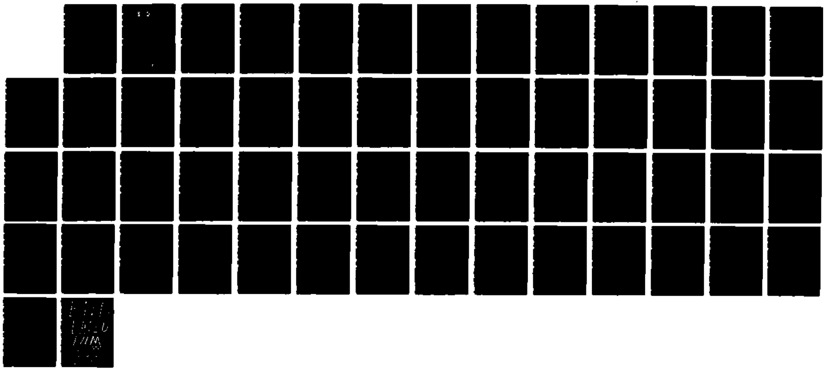
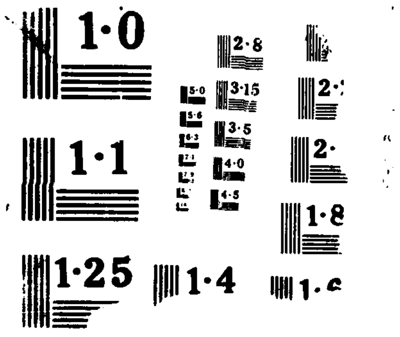
AD-A195 480   ARCHITECTURE STUDIES AND SYSTEM DEMONSTRATIONS FOR   1/1
OPTICAL PARALLEL PROCESSOR FOR AI AND NI(U) CALIFORNIA
UNIV SAN DIEGO LA JOLLA S H LEE 31 MAR 88
UNCLASSIFIED AFOSR-TR-88-0579 AFOSR-88-0022   F/G 20/6   NL




1·0
2·8
3·15
5·0
5·6
6·3
3·5
4·0
4·5
2·2
2·0
1·1
1·8
1·25
1·4
1·6

(2)

AD-A195 480

DTIC

REPORT DOCUMENTATION PAGE

DTIC FILE COPY

ELECTE JUN 2 9 1988 S D H

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S)		5. MONITORING ORGANIZATION REPORT NUMBER(S) AFOSR-TR- 88-0579	
6a. NAME OF PERFORMING ORGANIZATION University of California	6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION AFOSR/NE	
6c. ADDRESS (City, State and ZIP Code) LaJolla, CA 92093		7b. ADDRESS (City, State and ZIP Code) Bldg 410 Bolling AFB, DC 20332-6448	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION AFOSR/NE	8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER AFOSR-88-0022	
8c. ADDRESS (City, State and ZIP Code) Bldg 410 Bolling AFB, DC 20332-6448		10. SOURCE OF FUNDING NOS.	

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
61102 F	DARPA 6150	00	

11. TITLE (Include Security Classification) Architecture Studies and System Demonstrations of Optical Parallel Processor for AI and NI

12. PERSONAL AUTHOR(S) Professor Sing H. Lee

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
Semi-Annual	FROM 01/10/87 TO 31/03/88		

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB. GR.	

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

In solving deterministic AI problems the data search for matching the arguments of a PROLOG expression causes serious bottleneck when implemented sequentially by electronic systems. To overcome this bottleneck we have developed the concepts for an optical expert system based on matrix-algebraic formulation, which will be suitable for parallel optical implementation. The optical AI system based on matrix-algebraic formation will offer distinct advantages for parallel search, adult learning, etc.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL GILES	22b. TELEPHONE NUMBER (Include Area Code) (202) 767-4932	22c. OFFICE SYMBOL NE

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

SECURITY CLASSIFICATION OF THIS PAGE

AFOSR-TR- 88-0579

Semiannual Technical Report*

for

Architecture Studies and System Demonstrations of Optical Parallel

Processor for AI and NI

April 15, 1988

Sponsored by

Defense Advanced Research Projects Agency

DARPA Order No. 6150, Code 7D10

Monitored by AFOSR Under Grant No. AFOSR-88-0022

Grantee

The Regents of the University of California

University of California, San Diego

La Jolla, CA 92093

Principal Investigator:

Sing H. Lee

(619) 534-2413

Program Manager:

Dr. C. L. Giles

(202) 767-4931

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

88 6 29 078

Summary

In solving deterministic AI problems the data base search for matching the arguments of a PROLOG expression causes serious bottleneck when implemented sequentially by electronic systems. To overcome this bottleneck we have developed the concepts for an optical expert system based on matrix-algebraic formulation, which will be suitable for parallel optical implementation. The optical AI system based on matrix-algebraic formulation will offer distinct advantages for parallel search, adult learning, etc. (For more details see Sect. 2 and attached reference 1).

To optically solve the AI problems which involve multiple arguments we have studied optical architectures to implement vector-tensor and matrix-tensor multiplication. A matrix-tensor multiplier can provide interconnections between 2-D arrays of processors and 2-D arrays of memory cells. Conventionally, a matrix-tensor multiplier is based on space multiplexing, therefore, requiring extremely large space-bandwidth product from the optical channel. We have devised a matrix-tensor multiplier based on random phase coding and multiplexing, and evaluated and compared the performance characteristics of the random phase coded matrix-tensor multiplier with those of a conventional one. The comparison shows that there exists a trade-off between the dynamic range requirements of the system and the space-bandwidth product required by the optical channel, and that the phase-coded matrix tensor multiplier will reduce the space bandwidth product requirements for reasonable dynamic range requirements. (For more details on the matrix-tensor multiplier performances see Sect. 3 and Reference 2).

In the development of programmable interconnect for optical AI and NI systems, we have investigated the storage properties of various photorefractive crystals. We have found that in SBN:60 crystals the available charge carriers may be very efficiently used to store a large number (e.g., ~ 450-600 patterns) of interconnection patterns. (For more details see Sect. 4.)

DTIC
COPY
INSPECTED
6

r
☑
☐
☐
Ju[illegible]ation
By
Distribution/
Availability Codes

Dist	Avail and/or Special
A-1	

1. Introduction

During the last six months we have been studying the existing parallel computing architectures for Artificial Intelligence (AI) and Neural Intelligence (NI) in order to advantageously apply optical interconnects and to design more flexible systems based on space-time tradeoffs. We have been investigating different relevant topics such as the paracomputer, shared and distributed memory, reconfigurability of optical interconnect, routing of optical signals, parallel algorithms for sorting and vector-matrix multiplications, etc.

In the next section we will briefly describe the new approach for mapping AI problems on optical processing architectures, where we show that typical AI operations such as search and unification can be performed in parallel using optical processing by devising a new matrix algebraic formulation. In Sect. 3 we will describe a new approach in implementing programmable optical interconnect by employing a matrix-tensor multiplier based on random phase coding and photorefractive crystals. In Sect. 4 we will discuss the results of our studies on storage properties of photorefractive crystals and their use in programmable architectures for AI and NI applications.

2. Optical Expert System Based on Matrix-algebraic Formulation

The data base search for matching the arguments of a PROLOG expression encountered in solving deterministic AI problems (such as those encountered in expert systems) often causes serious bottlenecks, when it is performed serially by sequential electronic machines. The reasons of this bottleneck arise from the exponential nature of the serial search process, which can be alleviated by parallel search. Several optical architectures exist for performing matching operations in parallel. Perhaps one of the most time efficient optical architecture for parallel matching is that of the vector-matrix multiplier, whose input can be the query vector and the matrix can be the fact matrix. Hence, in our matrix algebraic formulation each fact relating to two arguments will be encoded into a binary matrix. Different facts relating to different pairs of arguments will be encoded into different binary matrices. New facts can be generated by performing matrix-matrix multiplications and 2-D logic operations. Therefore, optical AI systems based on matrix-algebraic formulation offer distinct advantages for adult learning over serial PROLOG based expert systems. More details about optical expert systems based on our new formulation can be found in the attached reference 1.

Promising as the matrix-algebraic formulation may be, we recently found also that when the number of objects that are related by the same fact increases, the space bandwidth product of the optical vector-matrix multiplier will also have to be increased and may become a limiting factor. To solve those AI problems whose facts contain more than two arguments we have been studying optical architectures for vector-tensor and matrix-tensor multiplications.

3. Matrix-tensor Multiplication by Random Phase Coding

A matrix-tensor multiplier can be employed to provide interconnections between a 2-D array of processors and a 2-D array of memory cells in similar ways as a vector-matrix multiplier, providing interconnections between 1-D arrays. To reduce the space-bandwidth product requirement of the matrix-tensor multiplier we studied the incorporation of random phase codes into the matrix and tensor components. We evaluated the performance characteristics of the random phase coded matrix-tensor multiplier and compared them to those of a conventional one based on space multiplexing (see Figs. 1, 2). This comparison shows that there exists a trade-off between the dynamic range requirements of the system and the space-bandwidth product required by the optical channel. For reasonable dynamic range requirements the phase coded approach to matrix-tensor multiplication can reduce the space-bandwidth product requirements. More details on our phase-coded matrix-tensor multiplier can be found in the attached reference 2.

4. Programmable architectures

A parallel processing system with programmable architectures for AI and NI will require storage of large capacity to store the interconnection patterns. In our attempt to develop such storage systems we used the experimental setup of Fig. 3 to study the storage properties of various photorefractive materials. We investigated the dependence of the write and erase time response asymmetry (see Fig. 4) on different parameters (e.g., applied electric field, input intensity, crystal thickness, etc.). The photorefractive response time for both writing and erasing a diffraction grating in a 3mm thick crystal of SBN:60 as a function of applied electric field is shown in Fig. 5. The ratio (R) of the photorefractive erase time over write time is plotted in Fig. 6 as a function of an externally applied electric field. We found that an asymmetry of almost 30 was reached at 8 kV/cm. This asymmetry could be further enhanced by increasing the input writing beams intensity and the thickness of the crystal. The larger the asymmetry is, the larger is the number of holograms that can be stored in the the photorefractive crystals. In our preliminary experiments we have stored 30-40 planes of 2-D information using a crystal providing a ratio R = 2. Assuming that this asymmetry results from a more efficient use of the available charge carriers, we expect that the memory capacity of such crystals can be increased to 450-600 planes of information.

5. Conclusions and Further Directions

During the past six months we have devised a new optical expert system based on matrix algebraic formulation and suitable for parallel optical implementation. We are continuing the study of our matrix-algebraic formulation for AI by finding the relations between the fact matrices and rules so that sparse or infrequently used fact matrices can be incorporated into rules to save memory space. This approach will also be compared to other parallel AI knowledge base systems that use matrix formulations.

We have also developed a new approach to implement programmable optical interconnect by employing a matrix-tensor multiplier based on random phase coding and compared its performances to those of a conventional one based on space multiplexing. We are presently implementing experimentally the phase coded matrix-tensor multiplier using dichromatic gelatin and photorefractive crystals.

In our development of optical storage for programmable (interconnect) architecture we have studied the storage properties of different photorefractive materials and predict that in SBN:60 the available charge carriers can be very efficiently used, therefore allowing very large storage capacity (i.e., ~ 450-600 inteconnection patterns). We are currently investigating the maximum storage capacity of photorefractive SBN:60 experimentally. Concurrently, we are conducting a study on the reprogrammability of interconnection networks (i.e., erase some of the stored interconnection pattern and write some new ones). Results of these and other studies will be provided in our future reports.

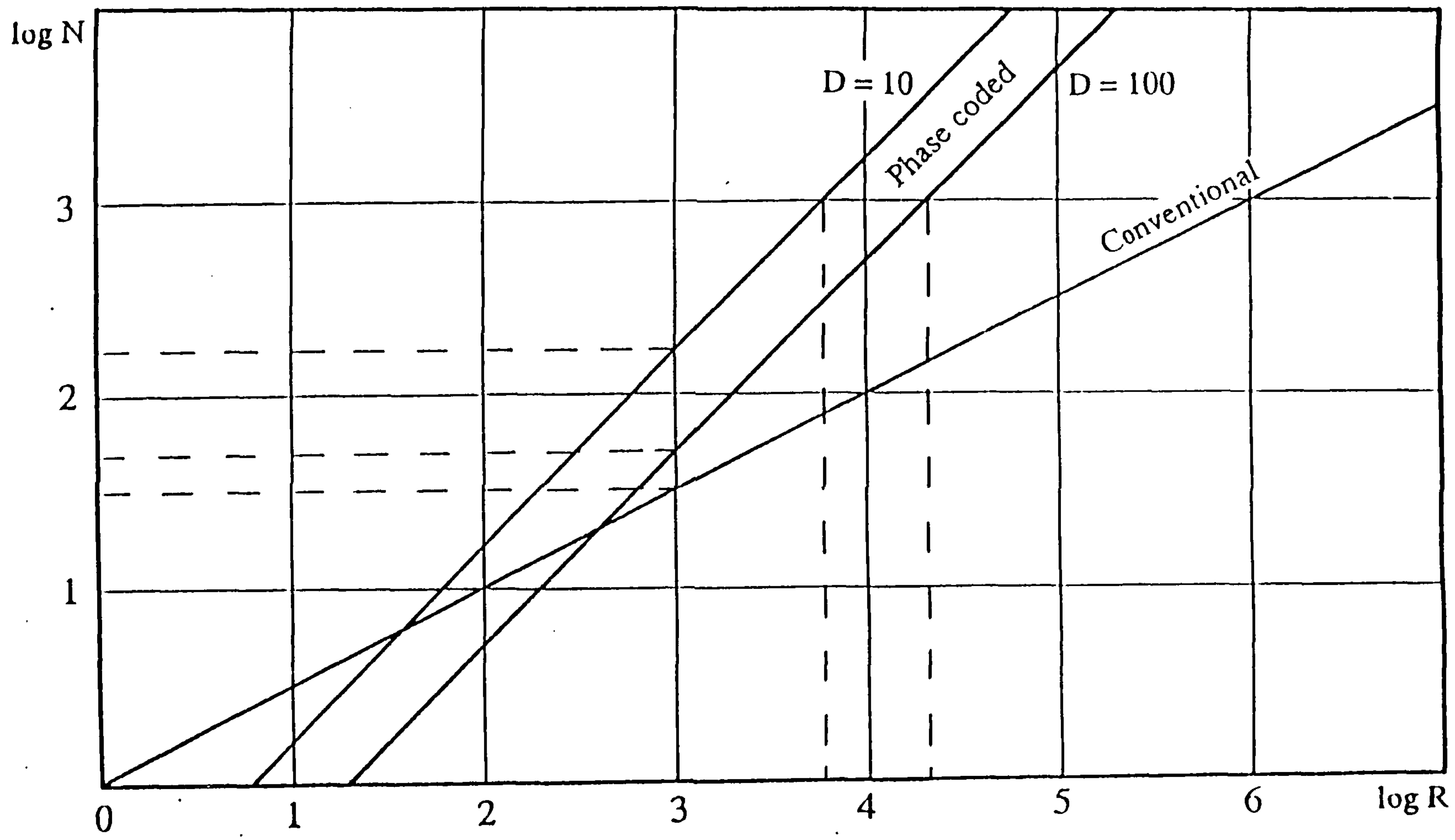


Fig. 1. Comparison of phase coded and conventional matrix-tensor multiplication using planar material. R^2 is the available space bandwidth product (SBP) provided by the real-time nonlinear material (RTNM); N^2 is the size of the input/output 2-D array. To operate on a matrix of 1000×1000 the conventional method will require a SBP of $10^6 \times 10^6$ from the RTNM, while the phase coded M-T multiplier will require only a SBP of $2 \cdot 10^4 \times 2 \cdot 10^4$ for D = 100 and $6 \cdot 10^3 \times 6 \cdot 10^3$ for D = 10 from RTNM. Alternatively, given a SBP of 1000×1000 for the RTNM, the conventional M-T multiplier can accomodate a matrix of 32×32, while the phase coded M-T multiplier will be able to accomodate a 50×50 matrix for D = 100 and 160×160 matrix for D = 10.

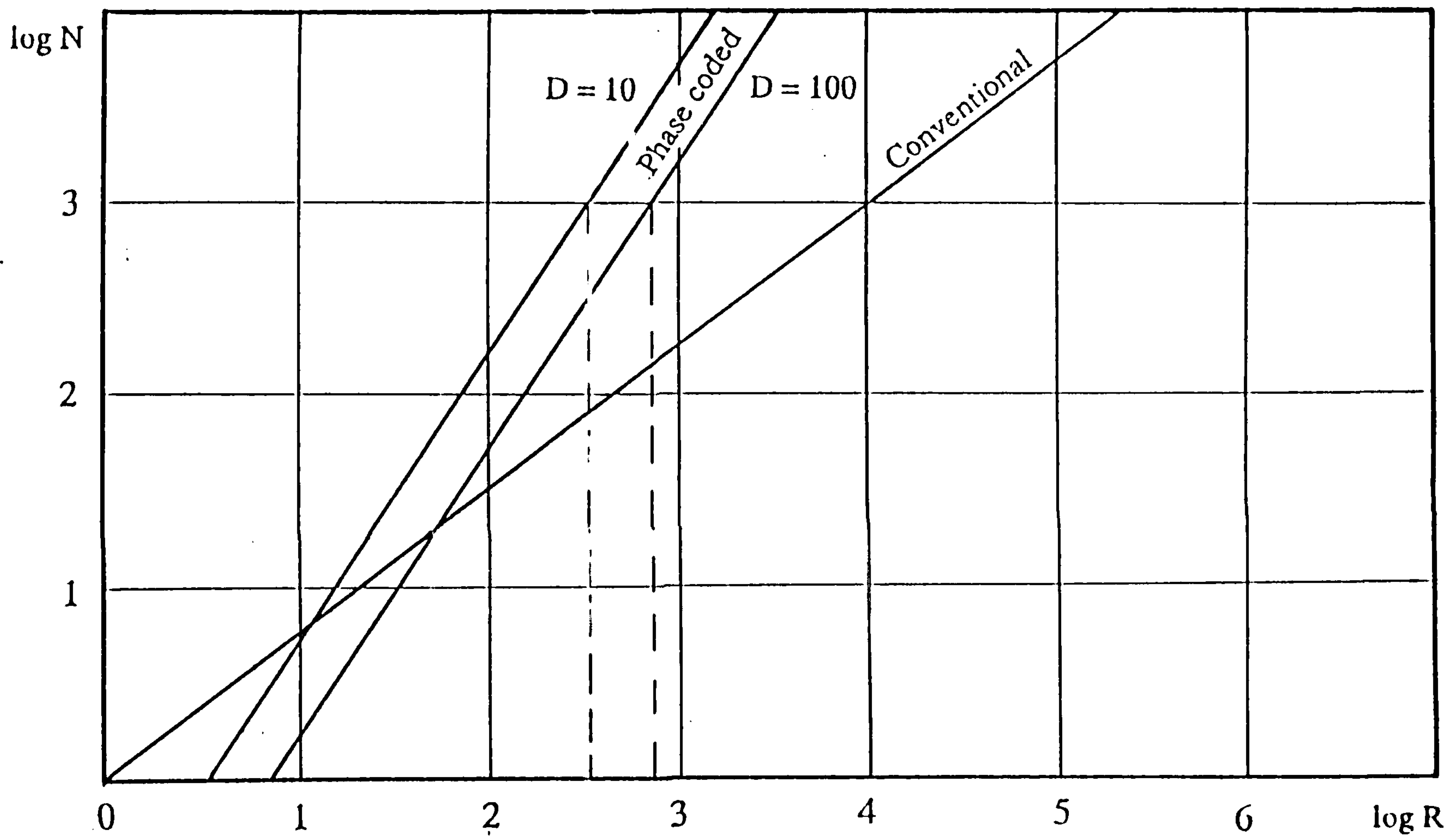


Fig. 2. Comparison of phase coded and conventional matrix-tensor multiplication using volume material. R^3 is the available space bandwidth product (SBP) provided by real-time nonlinear material (RTNM); N^2 is the size of the input/output 2-D array. To operate on a matrix of 1000×1000 arrays a conventional method will require a SBP of $10^4 \times 10^4 \times 10^4$ from RTNM, while the phase coded multiplier will require a SBP less than $10^3 \times 10^3 \times 10^3$ for D = 100 and a SBP of less than 400×400×400 for D = 10 from RTNM.

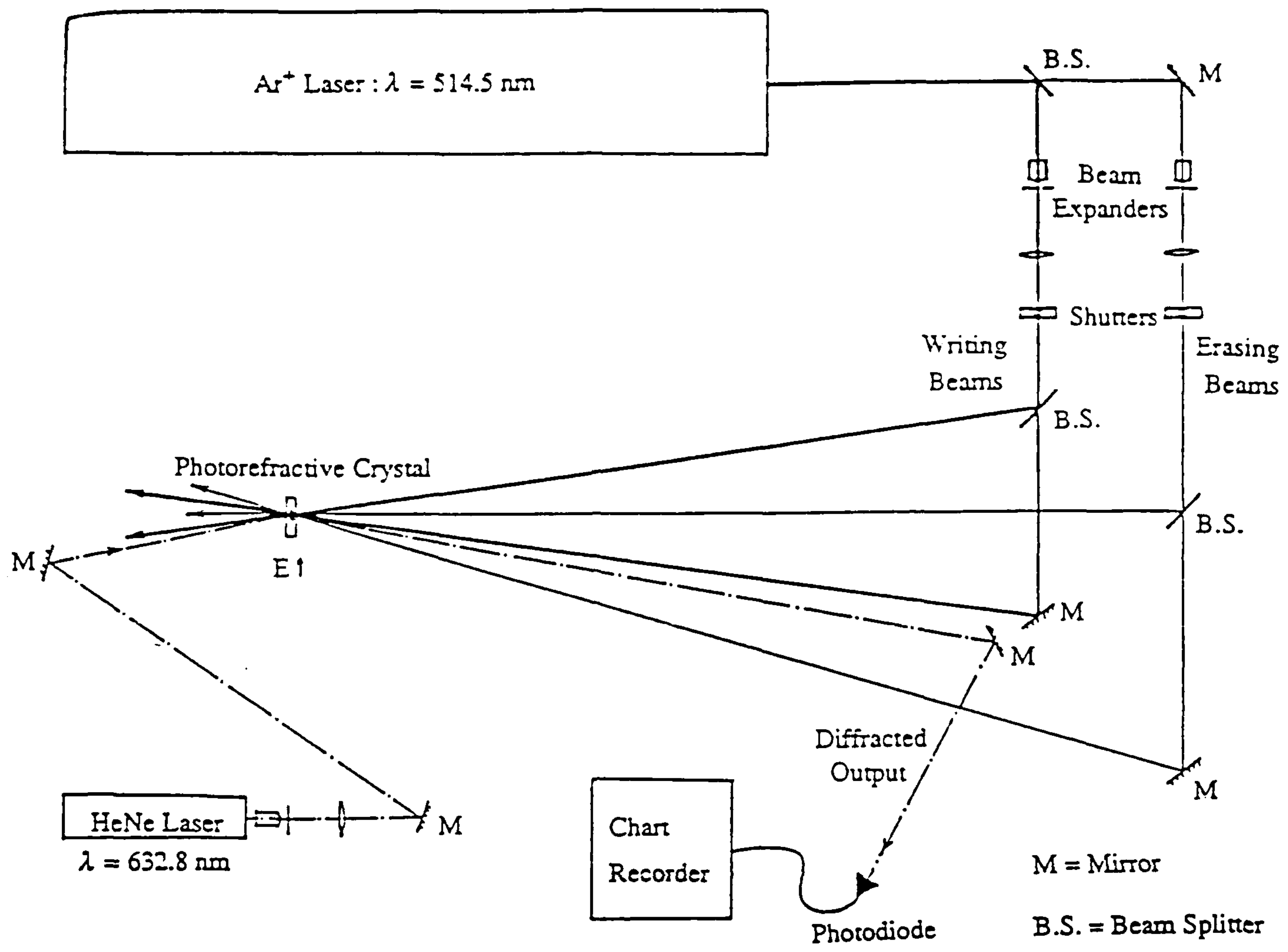


Figure 3. An experimental set-up designed to measure the write time and erase response time characteristics of photorefractive materials. the first pair of plane waves from the argon laser is used to write a grating in the photorefractive crystal. A HeNe beam which is Bragg-matched to this grating is then partially diffracted into a photodiode, whose output is stored in a chart recorder. This way the grating's presence can be continuously monitored. The grating is then erased by blocking the first beam pair and immediately illuminating the crystal with a second pair of beams from the argon laser. A new grating, for which the HeNe beam is not Bragg matched, is written in the crystal, eventually replacing the first grating. The write and erase times are directly measured. The ratio of the erase time to the write time, R, can then be calculated.

The intensities of each of the four argon laser beams are unifrom across the crystal surface and equal to one another to within 5%. The crystal is held between two electrodes and immersed in high voltage oil to prevent arcing between the electrodes. The field is applied parallel to the grating vector of the primary grating. Reversing the direction of the field did not significantly affect the result. The face of the crystal was normal to the primary beam pair. Rotating the crystal by up to ten degrees in either direction did not significantly alter R, although the overall diffraction efficiency was reduced. The time required to switch between beam pairs much less than the response times observed. The intensity of the HeNe beaem used to read the grating was small compared to the Ar writing beams, so that erasure of the grating by the reading beam was negligible.

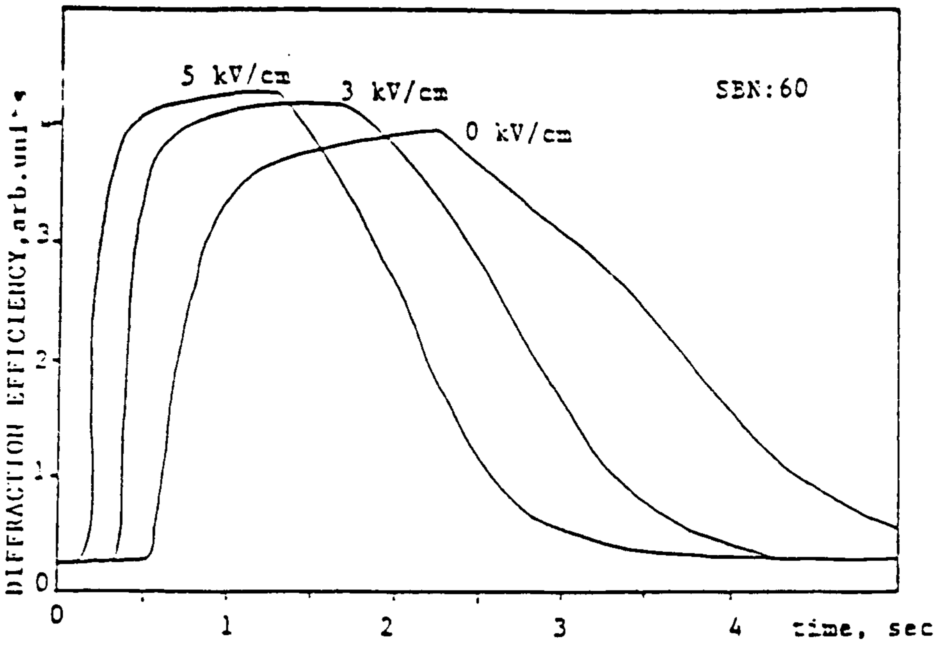


Figure 4. A typical write-erase time cycle of a 3mm thick SBN:60 crystal, measured at different values of externally applied electric field parallel to the grating wavevector. The asymmetry in write and erase time is enhanced by the externally applied electric field.

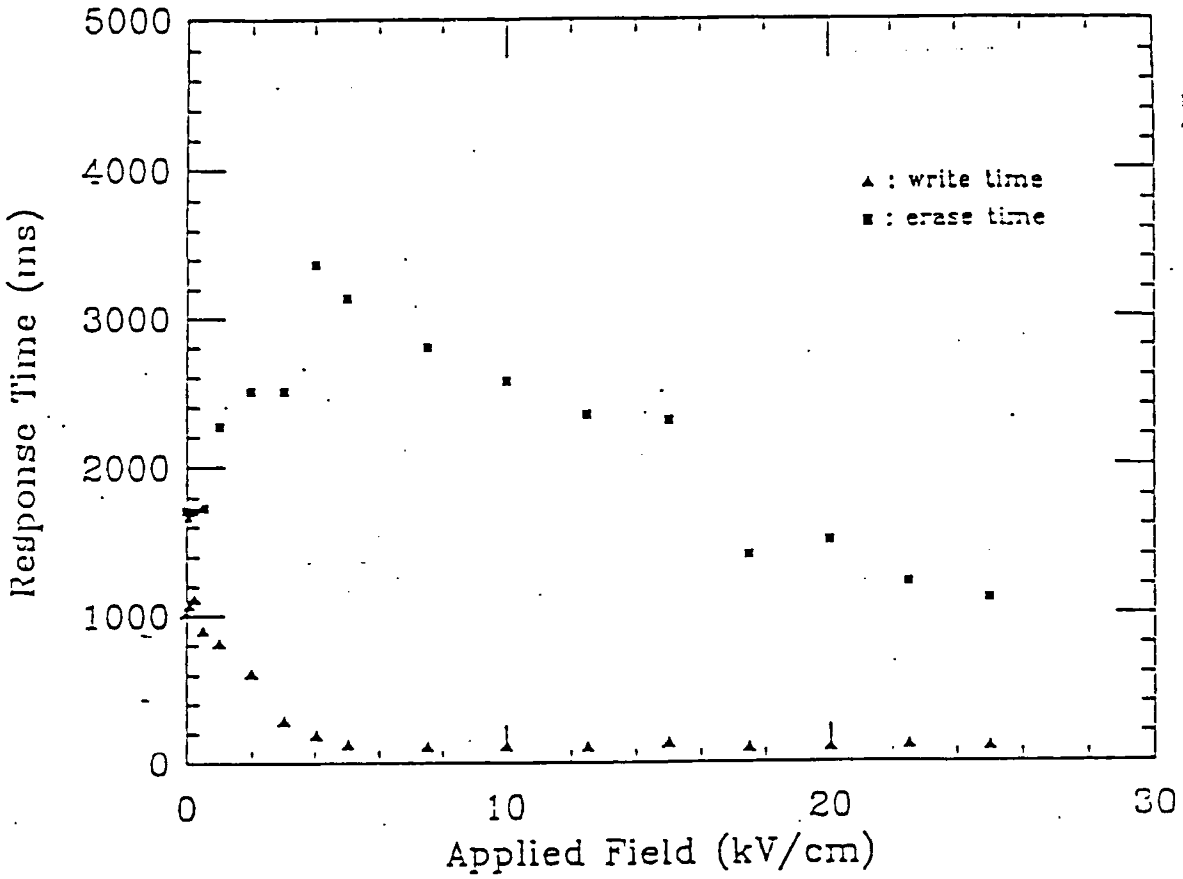


Figure 5. The photorefractive response time for both writing and erasing a diffraction grating in a 3 mm thick crystal of SBN:60 is plotted as a function of externally applied electric field. The intensity of each of the light beams was 750 mW/cm2. The erase time increases to a peak at 5 kv/cm, then decreases, while the write time decreases monotonically with applied field.

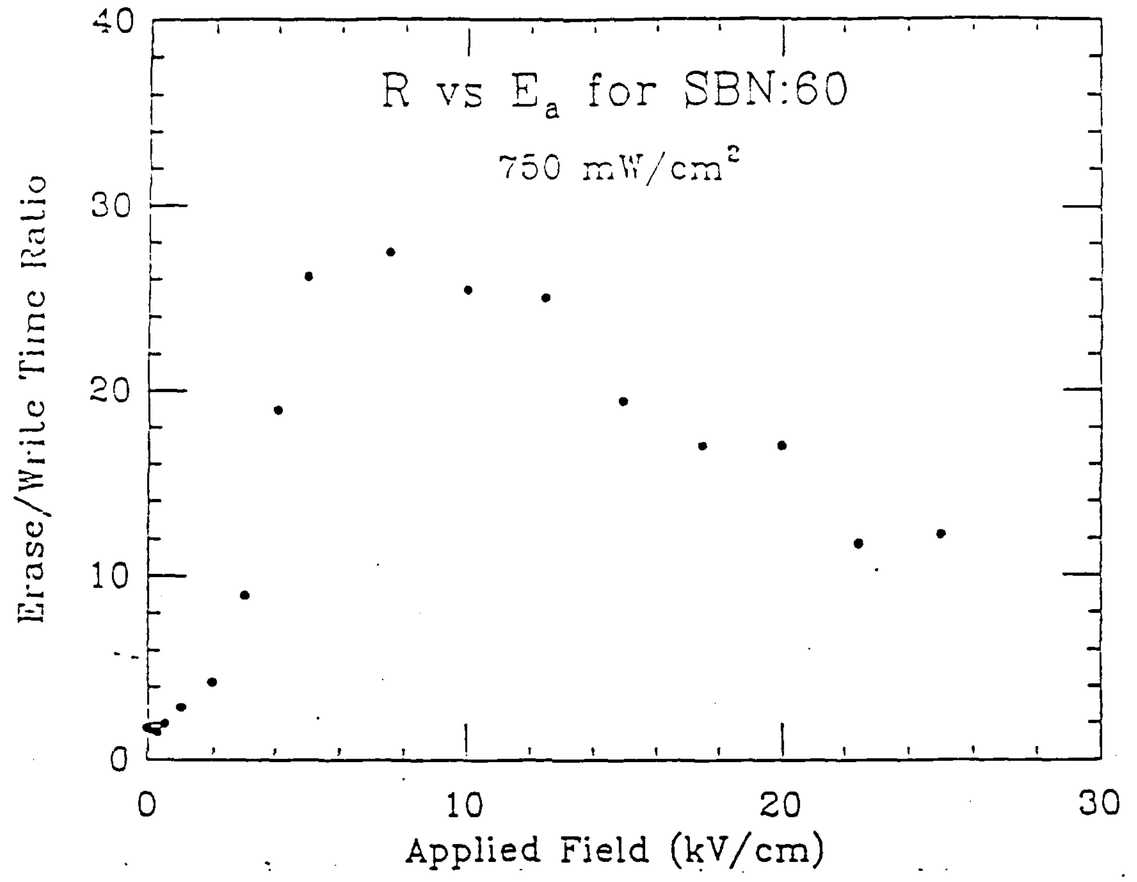


Figure 6. The ratio, R, of the photorefractive erase time over the write time is plotted as a function of applied field for a 3 mm thick crystal of SBN:60. A peak value of almost thirty was reached at 8 kV/cm. This asymmetry should result in a corresponding increase in the number of image planes which can be simultaneously stored in the volume of the crystal. Date Started: 5-24-88

REF. 1

An Optical Expert System Based on Matrix-algebraic Formulation

Jack Y. Jau, F. Kiamilev, Y. Fainman, S. Esener and Sing H. Lee

Department of Electrical and Computer Engineering
University of California at San Diego
La Jolla, California 92093

Abstract

This paper describes an expert system paradigm based on matrix algebra. The knowledge base of the expert system is stored in binary matrices, while the learning and the inference processes are done by matrix algebra operations. This method is highly parallel and can take full advantage of the inherent parallelism and connectivity of optics. An opto-electronic architecture that implements this system is presented. In addition, the method is compared to the sequential search methods written in the programming language Prolog in order to illustrate their differences and commonalities.

1. Introduction

Artificial intelligence (AI) is a field concerned with the simulation of intelligent behavior[1]. Expert systems have been one of the most successful applications of AI[2]. Expert systems encode knowledge in the knowledge base and perform intelligent interactions with it. Therefore, knowledge representation has become a fundamental problem in AI. One approach to knowledge representation is the use of semantic networks (see Figure 1 for an example), where the relations (or attributes) between objects and values (e.g. Mary, football, etc.) are described by a directed graph consisting of nodes and labeled edges (e.g. Likes and Member). To store a semantic network, conventional electronic expert systems use a tuple ordered set (attribute, object, values) . Many of these expert systems are based on the production system paradigm[3-4], where facts and rules constitute a knowledge base. Facts define the relationships between objects, and rules define the procedures for deriving new relations from existing ones. Prolog is a popular AI language that implements the production system paradigm[5].

Conventional electronic expert systems sequentially search the knowledge base for the appropriate facts and rules in order to find solutions to a query. The use of these electronic systems to solve symbolic logic problems has been limited, due to the inefficiency of such massive searches. To support numerous AI applications, more powerful computing machines capable of performing a massively parallel search are needed. Parallel search allows an expert system to find solutions efficiently because all the potential solutions are considered simultaneously. Some systems that perform symbolic computing and inference utilizing the parallelism and connectivity of optics have recently been presented. For example, Eichmann and Caulfield[6] proposed an optically assisted expert system using optical spatial light modulators(SLMs) to store knowledge. Warde and Kottas[7] described two query-driven hybrid optical inference machines. One was based on the conventional matched-filter concept, which is similar to the optical correlograph system described by Willshaw and Longuet-Higgins[8]. The other used mapping templates to store the relationships between objects; conclusions to the query are inferred by applying these mapping templates to objects in the order prescribed by the rules. Szu and Caulfield[9] generalized the prior work by representing the knowledge in an associative memory matrix and storing data explicitly in a 2-D outer product matrix so that logic inference could be performed at an extremely fast rate.

In this paper, we attempt to formulate an expert system paradigm based on matrix algebra in a form which is more suitable for optical implementation. Key issues related to this expert system paradigm, including knowledge representation, learning capability and the inference engine will be discussed in Sections 2 and 3 respectively. A comparison of the proposed method with sequential search-based methods written in the programming language Prolog is provided to illustrate the differences and commonalities between them. We base our comparison on sequential Prolog because it is currently one of the most widely used programming language in the AI community, although parallel versions of the language are available. In Section 4, we will present (as an example) a maze search problem to show the parallel processing nature of the matrix encoding method. An optical architecture for the implementation of the proposed expert system paradigm is described in Section 5. Section 6 concludes the presentation.

2. Knowledge representation of facts and rules

Knowledge representation is a key issue in the design of a knowledge-based expert system. It will affect the architecture of the system as well as the efficiency of knowledge retrieval from the system. In this Section we describe a matrix encoding method for knowledge representation, where facts of relations among objects are encoded in binary matrices, and rules are manipulated in matrix algebraic form.

2.1 *Matrix encoding of facts*

Let R_x^n and R_y^m be two sets of objects,

$$R_x^n = \left\{ x_1, x_2, \ldots, x_n \right\}, \quad R_y^m = \left\{ y_1, y_2, \ldots, y_m \right\} \tag{1}$$

where x_i and y_j are objects in R_x^n and R_y^m, respectively. If the objects in R_x^n are related to the objects in R_y^m, then, their relationships can be described by a set of facts. In logic formulation, a *fact* with two arguments can be seen as a binary function of the two related objects; that is, if the relationship between x_i and y_j is true, then $fact(x_i, y_j) = 1$, otherwise $fact(x_i, y_j) = 0$. In our matrix encoding method, *facts* of the same relationship (i.e. all the tuples with the same attribute) are encoded in a binary matrix. If $fact(x_i, y_j) = 1$ (true) or $fact(x_i, y_j) = 0$ (false), then the matrix element $[F_{ij}]$, at the i-th row and the j-th column of matrix $F(R_x^n, R_y^m)$, is equal to 1 or 0, respectively; i.e.

$$[F_{ij}] = fact(x_i, y_j) = \begin{cases} 1 & true \\ 0 & false \end{cases} \quad x_i \in R_x^n, \; y_j \in R_y^m; \tag{2}$$

or

$$F(R_x^n, R_y^m) \equiv \begin{array}{c} \\ x_1 \\ x_2 \\ . \\ x_i \\ . \\ x_m \end{array} \begin{bmatrix} y_1 & y_2 & . & . & y_j & . & . & y_n \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & 1 & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \end{bmatrix}, \tag{3}$$

where $F(R_x^n, R_y^m)$ (or F for short) is called the fact matrix for a certain relation between R_x^n and R_y^m. For example, we illustrate in Figure 1 a semantic network, where a node defines either the object or its value, and an arrow indicates the relationship between object and value. Based on our proposed matrix encoding method, the semantic network of Figure 1 can be fully described by the matrices L (for relation *Likes*) and M (for relation *Member*),

$$L \equiv \begin{array}{c|cccc} & John & Mary & David & Ann \\ \hline John & 1 & 0 & 0 & 1 \\ Mary & 0 & 1 & 1 & 0 \\ David & 0 & 0 & 1 & 0 \\ Ann & 1 & 0 & 0 & 1 \end{array} \tag{4}$$

$$M \equiv \begin{array}{r|ccc|} & \textit{football} & \textit{baseball} & \textit{tennis} \\ \textit{John} & 1 & 0 & 0 \\ \textit{Mary} & 0 & 0 & 1 \\ \textit{David} & 0 & 1 & 0 \\ \textit{Ann} & 0 & 0 & 1 \end{array} . \tag{5}$$

It is useful to note that the *likes* relation matrix(L) is a square matrix and relates objects of the same set $R_y^4 = R_x^4 = \{$ John, Mary, David, Ann $\}$ to one another, while the *Member* matrix(M) is a rectangular matrix and relates objects in two different sets, $R_x^4 = \{$ John, Mary, David, Ann $\}$ and $R_y^3 = \{$ football, baseball, tennis $\}$, to one another.

2.2. *Matrix algebraic formulation of rules*

The other important element of a knowledge base in an expert system consists of rules, which can be used to generate facts for new relations from given or already known facts. We may consider, in general, that a rule is a function of facts. A new fact can be concluded, from the function, through a sequence of logic operations applied to the known facts. In the matrix encoding method, logic operations between relations are carried out by thresholding the matrix-matrix product, for the AND operation, and by thresholding the matrix-matrix sum, for the OR operation. Therefore in this paper, all the matrix algebra operations presented are followed by an implicit threshold operation. For example, given three matrices $A(R_x^n)$, $B(R_y^m)$ and $C(R_z^l)$ of known facts, if a rule says that "M is true if C is true or both A and B are true", then this rule can be formulated in matrix algebra form,

$$M(R_x^n, R_z^l) = C(R_x^n, R_z^l) + A(R_x^n, R_y^m) \cdot B(R_y^m, R_z^l), \tag{6}$$

where M is the fact matrix of a new relation. As another example, let us define the following rule:

If (x *Likes* y) and (y is a *Member* of the z team),

then x may like to *Watch* z game.

The new relation *Watch*(*W*) for x and z is true, if both relations *Likes*(*L*) for x and y and *Member*(*M*) for y and z are true. In matrix algebraic formulation, the fact matrix W can be generated from L in Eq.(4) and M in Eq.(5) by a matrix-matrix multiplication operation; i.e.

$$W = L \cdot M = \begin{bmatrix} 1\,0\,0\,1 \\ 0\,1\,1\,0 \\ 0\,0\,1\,0 \\ 1\,0\,0\,1 \end{bmatrix} \begin{bmatrix} 1\,0\,0 \\ 0\,0\,1 \\ 0\,1\,0 \\ 0\,0\,1 \end{bmatrix} = \begin{array}{l} \\ \textit{John} \\ \textit{Mary} \\ \textit{David} \\ \textit{Ann} \end{array} \begin{array}{c} \begin{array}{ccc} \textit{football} & \textit{baseball} & \textit{tennis} \end{array} \\ \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \end{array} \qquad (7)$$

It should be noted that all the new facts for the same relationship are generated simultaneously in parallel by a matrix-matrix multiplication, and are encoded in a single matrix, even though only some of the new facts may be needed to solve the problem. This is different from most conventional expert systems, where only the needed facts are generated one at a time.

2.3 *Comparison with Prolog*

A Prolog expert system stores and processes all the facts separately. Consider the facts of relationships *Likes* and *Membership* in the preceeding examples. The Prolog code for these facts is:

```
Likes(John, John)
Likes(John,  Ann)
Likes(Mary, Mary)
```

Likes(Mary, David)
Likes(David, David)
Likes(Ann, John)
Likes(Ann, Ann)
Member(John, football)
Member(Mary, tennis)
Member(David, baseball)
Member(Ann, tennis)

Only the facts for which the relationship is true are stored in the knowledge base. In contrast, in the matrix-algebra encoding method all the facts (true or false) of the same relation are stored in a single binary matrix. All the facts stored in the same matrix are later processed simultaneously.

The number of arguments in a fact can be arbitrary. For example, the fact "Gives(John, Mary, book)" has three arguments. In Prolog, the processing time increases with increasing number of arguments. In contrast, in matrix-algebra encoding method the number of arguments in a fact will determine the rank of a tensor, which stores the fact (e.g. two arguments determines a second rank tensor which is a matrix). Therefore, the matrix encoding is capable of operating on more than two arguments, but at the expense of an increase in space complexity.

In Prolog conclusions are derived from rules sequentially by asserting the left side of a rule whenever all the relations on the right side of the rule can be unified. For example, the rule for *Watch* given earlier translates into:

Watch(x,z) :- Likes(x,y), Member(y,z) ,

and a new fact *Watch* between x and z is asserted, if prolog find that both the facts $Likes(x,y)$ and $Member(y,z)$ are true for some objects x, y and z. This derivation is done by sequentially searching the knowledge base for every matching solution. In addition, the newly derived facts are usually not saved to avoid increasing the knowledge base and the search time to find the solution. In contrast, with the matrix encoding method all the conclusions are generated simultaneously and can be saved without extra overhead in search time as shown in Eq.(7).

3. Learning and the inference engine

In an expert system, inference is the process to find a solution to a query, while learning is the process of updating the knowledge base. Conventional electronic expert systems use sequential search with unification (e.g. pattern-matching process) to perform inference and learning. In this section a parallel scheme is described, based on the matrix encoding method that updates the knowledge base and answers queries.

3.1. *Learning: a dynamic knowledge base*

In a matrix-algebra based expert system, learning can be implemented simply by using rules, where new relations can be generated from known relations and facts in the knowledge base. As shown in Eq.(7), given known relations *Likes* (L) and *Member* (M), the new relation *Watch* is learned by performing matrix-matrix multiplication. An optical matrix-matrix multiplier can offer much help in speeding up the learning of such an expert system.

In special cases when an existing relation has to be updated,

$$L^{new} = L^{old} + (L^{old} \cdot G)^t \,, \tag{8}$$

a gain matrix of learning G is required as illustrated by the following example. The following rule

If x *Likes* "Mary", then "John" *Likes* x,

can be expressed in matrix algebraic formulation by

$$\begin{array}{c|cccc} & John & Mary & David & Ann \\ \hline John & 1 & 1 & 1 & 1 \\ Mary & 0 & 0 & 0 & 0 \\ David & 0 & 0 & 0 & 0 \\ Ann & 0 & 0 & 0 & 0 \end{array}^{t} = \begin{array}{c|cccc} & John & Mary & David & Ann \\ \hline John & 0 & 1 & 0 & 0 \\ Mary & 0 & 1 & 0 & 0 \\ David & 0 & 1 & 0 & 0 \\ Ann & 0 & 1 & 0 & 0 \end{array} \cdot G \quad (9)$$

On the right-hand side of Eq.(9) the matrix indicates that "x *Likes* Mary" and x could be John, Mary, David or Ann, while on the left-hand side of the equation the matrix indicates that "John *Likes* x". Therefore, the gain matrix G is needed to equate those two matrices. Alternatively,G can be found simply by putting "1" in the intersection of the second row and the first column, which corresponds to the objects "Mary" and "John" respectively; i.e.

$$G = \begin{bmatrix} 0\,0\,0\,0 \\ 1\,0\,0\,0 \\ 0\,0\,0\,0 \\ 0\,0\,0\,0 \end{bmatrix} .$$

After evaluating Eq.(8), using L^{old} from Eq.(4) and the gain matrix G from Eq.(9), we obtain a new fact matrix for the relation *Likes*; that is

$$L^{new} \equiv \begin{array}{c|cccc} & John & Mary & David & Ann \\ \hline John & 1 & 1 & 0 & 1 \\ Mary & 0 & 1 & 1 & 0 \\ David & 0 & 0 & 1 & 0 \\ Ann & 1 & 0 & 0 & 1 \end{array} \quad (10)$$

Comparing the new fact matrix given in Eq.(10) with the old fact matrix given in Eq.(4), it is clear that the relation *Likes* has been updated by a new fact "John *Likes* Mary".

3.2 *Inference engine*

The inference engine finds a solution to a query by retrieving information related to the object x_i from the knowledge base. If the information (e.g. fact) is not immediately available, the matching rules are tried in order to conclude the query. The above description can be summarized in short syntax form as

$$\text{solution} \leftarrow \text{query} (\underline{x}, \text{facts, rules}).$$

In the matrix encoding method, retrieving from the matrix F a certain fact related to the object x_i is achieved through the following vector-matrix multiplication

$$\underline{y} = \underline{x} \cdot F,$$

where $\underline{y}$ ($=<y_1,y_2,..,y_m>$) is the solution vector, and $\underline{x}$ ($=<x_1,x_2,..,x_n>$) is the query vector. Both $\underline{x}$ and $\underline{y}$ are binary (0 or 1) vectors. If F encodes the facts of relationship between R_x^n and R_y^m, each element of $\underline{x}$ corresponds to one object in R_x^n, while each element of $\underline{y}$ corresponds to one object in R_y^m. The i-th element of the query vector $\underline{x}$ is set to indicate that x_i is the object of interest. Thus, if the j-th element of the solution vector $\underline{y}$ is set after the vector-matrix multiplication of $\underline{x}$ and F, the relationship between x_i and y_j is then true.

However, in most cases the fact is not immediately available in the knowledge base, and a rule must be employed in order to conclude a query. Let us consider an example of performing inference in order to answer the query:

What game may "John" like to *Watch* ?

In this particular case, the query vector is

$$\underline{x} = \begin{array}{cccc} John & Mary & David & Ann \\ <1 & 0 & 0 & 0> \end{array}' \tag{11}$$

and we inquire facts from relation *Watch*, i.e. the matrix W. The query can be concluded by a simple vector-matrix multiplication, $\underline{x} \cdot W$. However, if the fact matrix W is not yet in the knowledge base, the rule given in Eq.(7) must be called to conclude the query; i.e.

$$\underline{y} = \underline{x} \cdot W = \underline{x} \cdot L \cdot M = \begin{array}{ccc} football & baseball & tennis \\ [\,1 & 0 & 1\,] \end{array}' \tag{12}$$

The $\underline{y}$ vector indicates that the conclusion to the query is:

John may like to watch football and tennis games.

3.3 *Comparison with Prolog*

It should be noted that learning is done by rules. In the matrix-algebra based expert system all the learned facts are stored. Therefore, one pays a price in space (memory storage) but gains in processing speed. Adding new relations may increase the matrix size, without increasing the processing time. In contrast, in Prolog the new facts are usually not saved. If all the newly derived facts were saved in Prolog, the size of the knowledge base would increase. This would result in a significant increase in processing time.

Prolog is a backward reasoning system, which starts with the goal (query) and tries to work backwards to satisfy the goal. It solves a problem by sequential search and pattern-matching. For instance, to answer the query ?–*Watch*(*John*,*y*), i.e. what game John may like to watch?, Prolog searches the knowledge base (facts and rules) for the predicate *Watch* and matches the arguments step by step. Prolog also uses backtracking to find all possible solutions to the problem. This means that if the pattern-matching fails as Prolog searches through its knowledge base, then it automatically backtracks to its previous step, resulting in a depth-first type of search. Therefore, in Prolog the more arguments and rules that are used, the longer it takes to solve the problem. In contrast, the matrix algebra based expert system generates all the solutions to the problem at one time, even though some of the solutions are irrelevant to the goal. The parallel nature of optics eliminates the need for backtracking through the knowledge base.

4. Case study: Searching

To illustrate the differences, in terms of knowledge representation and searching algorithms, between Prolog encoding and the matrix encoding methods, let us consider the problem of searching through the maze shown in Figure 2(a). The simplest Prolog program for solving this maze is

```
DR(A,B).
DR(B,E).
DR(B,C).
DR(D,E).
DR(C,D).
DR(E,F).
DR(E,G).
```

```
GO(X,X).
GO(X,Y) :- DR(X,Z), GO(Z,Y).
```

where DR defines the door between two rooms, and GO(X,Y) is a rule that attempts to find a path from room-X to room-Y. To find if there is a path from room-A to room-G, a query ?- GO(A,G) is presented to Prolog. Note that this is equivalent to searching for a path between nodes A and G in the graph shown in Figure 2(b). Prolog would use sequential searching with backtracking to find this path. Accordingly, the time to find the path increases with the number of rooms and doors in the maze. The more rooms and doors there are, the longer it will take for Prolog to find the path. Finding the optimum (e.g. shortest) path will require even more time because there could be multiple solutions. In addition, the above Prolog program may get into an infinite loop (e.g. B-C-D-E-B-C-D-E-...). To avoid the looping problem, one can keep a list of rooms visited so far and avoid visiting the same room twice. This list will also give us the path from room-A to room-G.

In the matrix encoding method, the maze is encoded in a matrix D,

$$D = \begin{array}{c|ccccccc|} & A & B & C & D & E & F & G \\ \hline A & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ B & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ C & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ D & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ E & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ F & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ G & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{array} .$$

A recursive rule for GO is formulated as an iterative vector-matrix multiplication:

$$\underline{x}_A^{(k)} = \underline{x}_A^{(k-1)} \cdot D ,$$

To indicate that the search starts from room-A, the initial query vector would be $\underline{x}_A^{(0)} = < 1\ 0\ 0\ 0\ 0\ 0\ 0 >$. During the iterations, the following vectors are generated:

$$\begin{array}{lccccccccc}
 & & A & B & C & D & E & F & G & \\
\underline{x}_A^{(0)} = & < & 1 & 0 & 0 & 0 & 0 & 0 & 0 & >, \\
\underline{x}_A^{(1)} = & < & 1 & 1 & 0 & 0 & 0 & 0 & 0 & >, \\
\underline{x}_A^{(2)} = & < & 1 & 1 & 1 & 0 & 1 & 0 & 0 & >, \\
\underline{x}_A^{(3)} = & < & 1 & 1 & 1 & 1 & 1 & 1 & 1 & >.
\end{array}$$

Each output vector in the above sequence indicates the rooms that can be further reached from room-A. For example, the vector $\underline{x}_A^{(2)}$ indicates that we can get into rooms B, C and E from room-A because the vector elements which correspond to these rooms are set to "1". Since the vector element associated with room-G is set in $\underline{x}_A^{(3)}$, we conclude that there is a path between room-A and room-G. The superscript of $\underline{x}_A^{(3)}$ indicates that it takes three steps to reach room-G from room-A. The time needed to find an answer to a query in this search problem is independent of the number of doors, and the maximum number of vector-matrix multiplications (i.e. in the worst case) is proportional only to the number of rooms in the maze. Note that with this approach, an infinite loop will never occur. For an arbitrary graph, this search method always gives the shortest distance between two nodes. But the actual path between these two nodes is not provided. If the path information is desired, we have to actually walk down the graph to find the path, and this becomes a sequential search problem.

The above example illustrates the approach employed by the matrix encoding method for solving a search problem. It speeds up the search but requires extra memory space in order to store matrices. Although the matrix encoding method may not have problems with run-time, it may run into combinatorial explosion in space when dealing with huge search problems. There is essentially a trade-off between space and time requirements in solving these problems.

5. Optical Architecture

In the above sections, we have demonstrated that solutions to AI expert systems could be formulated in terms of simple binary matrix operations. These operations include vector-matrix and matrix-matrix multiplication, as well as some logic operations.

The information flow in matrix-algebraic expert systems for a sample query is shown in Figure 3. The solution $\underline{y}$ is formed by consecutive vector-matrix multiplications. The fact matrices that are loaded to the vector-matrix multiplier originate directly from the knowledge base or are computed by matrix-matrix multiplication from other fact matrices such as M2, M3 and M4. Therefore, an optical expert system architecture should include an optical memory for storing the fact matrices, optical hardware to perform matrix operations and an electronic micro-processor to control the information flow (Figure 4). The microprocessor also stores the rules in the form of matrix algebraic formulations and decides upon the strategy to answer the query. The output of the matrix-matrix multiplier is thresholded to perform the logic AND operation between two relations. The 2-D logic unit implements the logic OR operation between two relations by performing a matrix-matrix addition followed by a threshold operation.

To answer a query the microprocessor first forms an appropriate query vector which is then translated into optical form by an electronically addressed 1-D SLM. Second, the microprocessor determines the required linear algebra relations and their order of execution. Then, it loads the appropriate fact matrices from the optical memory and issues commands to the optical hardware to perform the appropriate operations. Operations that require iterations such as learning are performed by the optical hardware. The output of the optical hardware is then translated into electronic form by a 1-D detector array and the result is presented to the user. In certain problems the microprocessor can operate on the output of the optical hardware to decide on further action.

For example, to answer a query given by

$$\underline{y} = \underline{x} \cdot (C + A \cdot B),$$

the fact matrix $C + A \cdot B$ has to be generated before performing the vector-matrix multiplication. The microprocessor first loads the matrices A and B from the 3-D memory into the matrix-matrix multiplier (see Figure 4) and computes the product $A \cdot B$, which is then thresholded and stored in the 3-D memory temporarily. Next, the 2-D logic array performs the OR operation on matrices C and $A \cdot B$ that was recalled from the 3-D memory. The result which is the desired fact matrix $C + A \cdot B$ is again stored in the 3-D memory. The microprocessor then forms the query vector x and loads it into the 1-D SLM, while the fact matrix $C + A \cdot B$ is simultaneously loaded from the 3-D memory into the vector-matrix multiplier. Finally, the answer to the query is formed at the output of the multiplier and detected by a 1-D detector array and analyzed by the microprocessor.

6. Conclusions

This paper described an expert system paradigm based on a matrix-algebraic formulation and a potential opto-electronic architecture to implement this new paradigm. Facts of relations are encoded in matrices or tensors, depending on how many objects the facts are related to. All the rules for generating new relationships and for learning are formulated in matrix algebraic form and are performed in parallel by matrix-matrix multipliers. The inference capability can be achieved simply by vector-matrix multipliers, making full use of the parallelism and connectivity of optics. However, applications that require facts with more than two arguments necessitate a matrix-tensor multiplier. The comparison of the proposed paradigm with sequential Prolog has shown significant differences between them in knowledge representation and inference. A comparison of our matrix-algebra based paradigm with a parallel Prolog running on a multi-processor will be the next step in our research. A maze search example was discussed in order to illustrate the parallel processing nature of the matrix encoding method, which offers significant gain in processing speed but pays a price in the space required to store large matrices.

Acknowledgement:

This research work is supported by the Defense Advanced Research Project Agency, administrated by the Air Force Office of Scientific Research under Contract No. 88-0022.

REFERENCE

[1] T. O'shea and M. Eisenstadt, *Artificial intelligence: Tools, techniques and applications,* Harper&Row Publisher, 1984

[2] R. Forsyth, *Expert systems: principle and case studies,* Chapman and Hall, 1984.

[3] N. J. Nilsson, *Principles of artificial intelligence,* Tioga Publishing Company, Palo Alto, Calif. 1980.

[4] T. R. Addis, *Designing knowledge-based systems,* Prentice Hall, 1985.

[5] W. F. Clocksin and C. S. Mellish, *Programming in Prolog,* Springer Verlag, 1984.

[6] G. Eichmann, H. J. Caulfield and I. Kadar, "Optical artificial intelligence and symbolic computing: an introduction," *Appl. Opt.* Vol.26, No.10, May 1987. pp. 1827-1828.

[7] C. Warde and J. Kottas, "Hybrid optical inference machine: architecture considerations," *Appl. Opt.* Vol.25, No.6, March 1986. pp. 940-947.

[8] D. J. Willshaw and H. C. Longuet-Higgins, "Associative Memory Models," *Machine Intelligence,* Vol. 5, 1970, p. 351.

[9] H. H. Szu and H. J. Caulfield, "Optical expert system," *Appl. Opt.* Vol.26, No.10, May 1987, pp. 1943 - 1947.

[10] G. Eichmann and H. J. Caulfield, "Optical learning (inference) machines," *Appl. Opt.* Vol.24, No.14, July 1985. pp. 2051-2054.

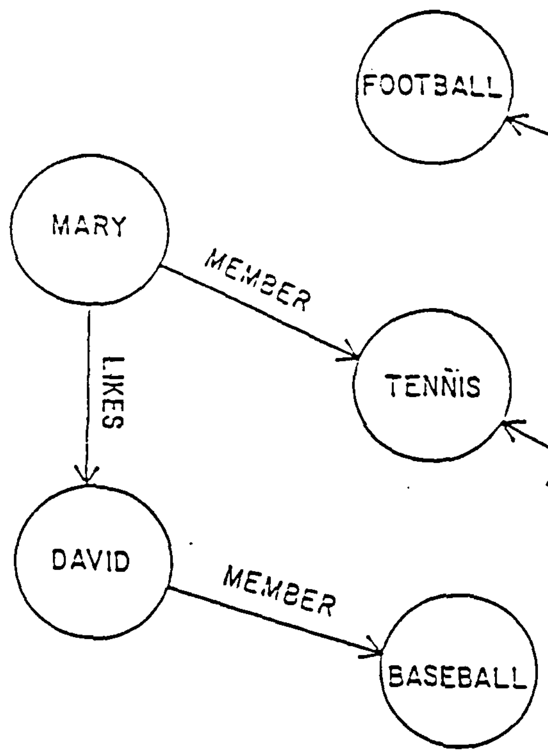
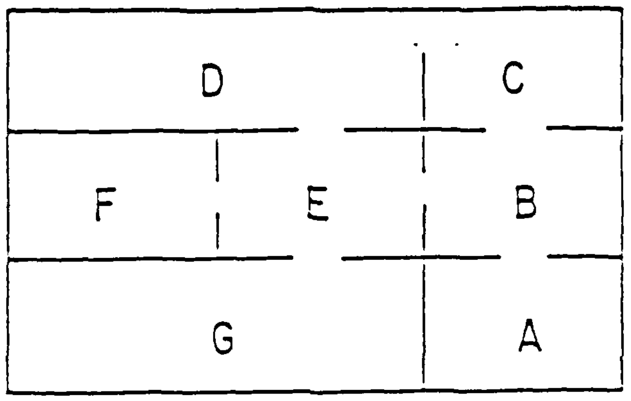
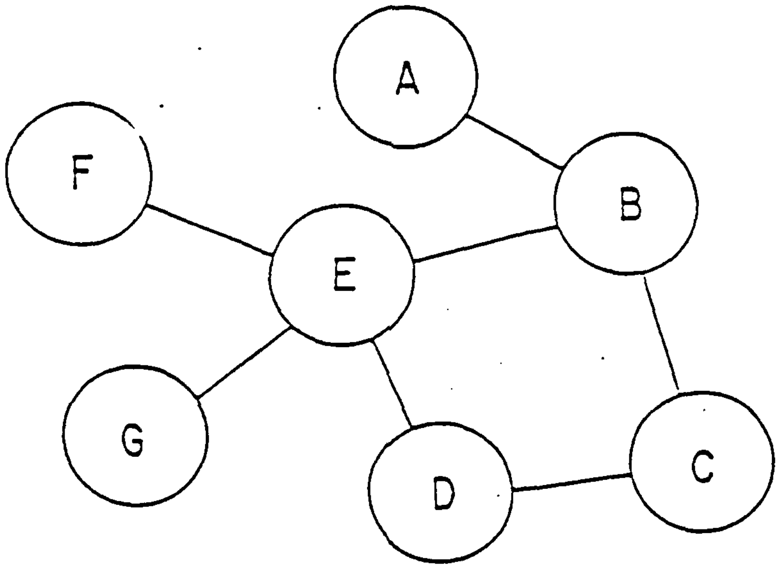


Figure 1. A semantic network: A node defines either the object (e.g. Mary) or its value (e.g. Tennis); A labeled edge (e.g. Member) indicates the relationship between object and value.



(a)



(b)

Figure 2. (a) A maze with rooms A, B,...,G and doors (indicated by broken lines) between rooms. (b) A graph used to represent the maze, where each node represents a room, and the connection between nodes indicates the door between rooms.

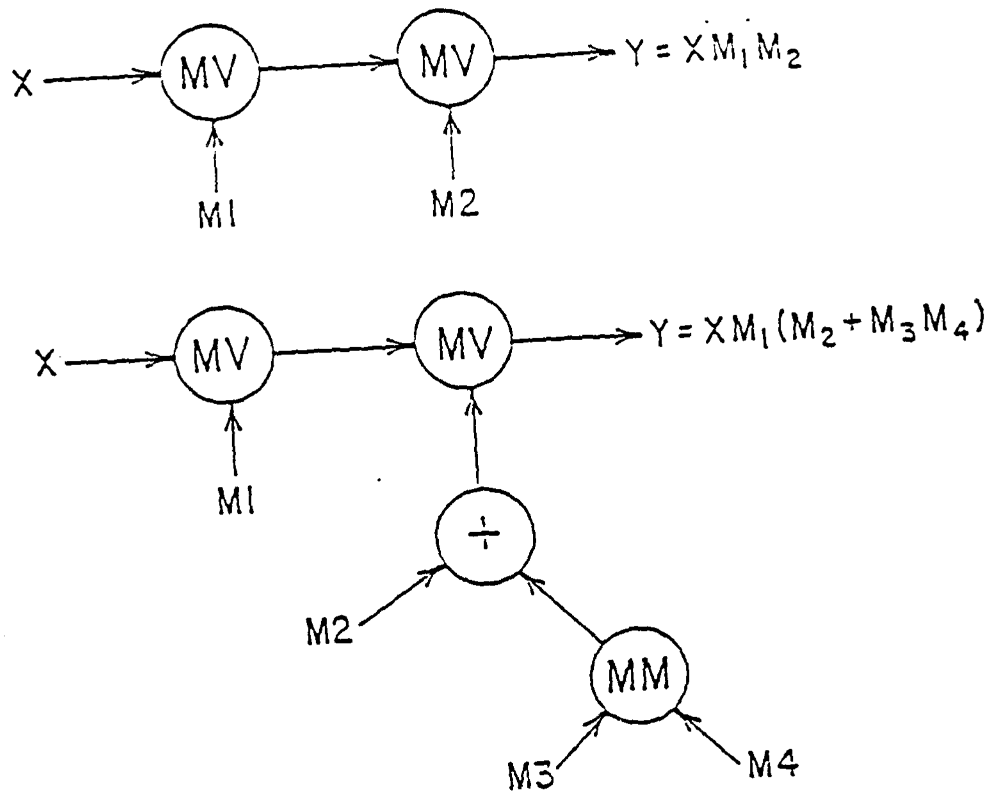


Figure 3. Schematic structures of information flow in matrix-algebraic expert system.

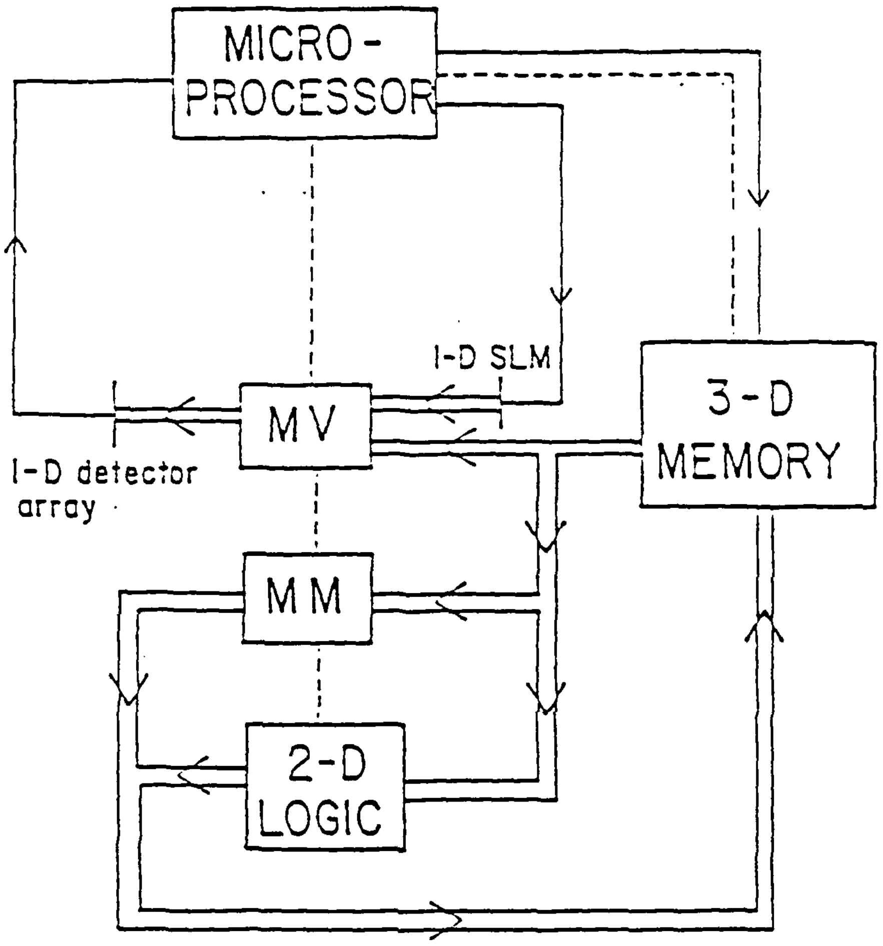


Figure 4. Architecture of an opto-electronic system for the implementation of matrix-algebraic expert system. The double line indicates the optical path; the solid line indicates the electronic signal path; the dashed line indicates the electronic control signal path.

REF. 2

Matrix-tensor Multiplication by Random Phase Coding

Y. Fainman and Sing H. Lee

Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093

D. Psaltis

Department of Electrical Engineering
California Institute of Technology
Pasadena, CA 91125

Abstract

A random phase coding technique is utilized to optically perform, in parallel, a matrix-tensor multiplication. This technique allows one to decrease the space bandwidth product of the optical system at the expense of a decrease in the system dynamic range. The analysis of this trade-off shows that this technique has advantages over the conventional techniques based on space multiplexing, and therefore, makes it possible to operate on large size 2-D arrays of data using currently available real time materials.

1. Introduction

Recently many areas of parallel optical computing have benefited from optical vector-matrix multipliers, such as, the performance of discrete Fourier transforms [1], analog computing [2,3], programmable optical interconnection [4], optical neural computing [5], etc. However, many applications require the manipulation of very large 2-D data arrays, which will essentially require the ability to perform matrix-tensor multiplication. Some attempts at solving this problem have been made in the past using space and spatial frequency multiplexing. These conventional matrix-tensor multipliers offer wide dynamic range; frequently the dynamic ranges are wider than those available from the Vidicons and the CCD-cameras (~ 6 to 8 bits) that are used in optical systems. Some optical computing architectures (e.g., digital computing and neural computing architectures) require a dynamic range of only 3 to 5 bits. On the other hand, they require an extremely large space-bandwidth product (SBP) for the optical channel.

In this letter we will describe a matrix-tensor multiplier based on a random phase coding process, which enables us to reduce the SBP requirements of the optical channel at the expense of the dynamic range or the signal-to-noise (*S/N*) ratio at the output. We will discuss the SBP and system dynamic range trade-offs in light of the fact that certain parallel optical processors have limited dynamic range requirements.

2. Matrix-tensor Multiplication Based on Space Multiplexing

To perform matrix-tensor multiplication between a 2-D input signal (matrix), g(x,y) and a fourth rank tensor, $\underset{\sim}{W}(x,y)$, we have to calculate the inner products of the function g(x,y) with each ℓm-th component of the tensor $\underset{\sim}{W}(x,y)$, where $\ell = 1, 2, ..., L$ and $m = 1, 2, ..., M$. The inner product of two functions can be determined from their correlation function evaluated at the origin. However, to perform matrix-tensor multiplication in parallel we have to compute in parallel LM such inner products. One obvious approach has been to spatially multiplex the components of the tensor in a 2-D plane (see Fig. 1a). This can be expressed as $\sum_{\ell=1}^{L} \sum_{m=1}^{M} W_{\ell m}(x-\ell d, y-md)$, where d is at least equal to the spatial extension of the components of the tensor $\underline{W}(x,y)$. The function

$$c(x,y) = g(x,y) \circledast \sum_{\ell=1}^{L} \sum_{m=1}^{M} W_{\ell m}(x-\ell d, y-md) \tag{1}$$

will consist of the spatially superposed correlation terms $g(x,y) \circledast W_{\ell m}(x-\ell d, y-md)$, where $\circledast$ denotes the correlation. At the coordinate points $(\ell d, md)$ this function provides the values of the desired inner products (Fig. 1b). Therefore, implementation of a matrix-tensor multiplier by space multiplexing will require an optical channel of SBP

$$SBP_{MT} = SBP_{M} LM, \tag{2a}$$

where SBP_{MT} is the SBP of the channel and SBP_{M} is the SBP of the signal. For example, if the matrix-tensor multiplier is used to fully interconnect an N×N array from the input to an N×N array at the output (L=M=N), a channel of

$$SBP_{MT} = SBP_{M}^{2} = N^{4} \tag{2b}$$

will be required. Such a large SBP_{MT} is required in order to (i) spatially multiplex the tensor components and (ii) separate the value of the inner products at coordinate point $(\ell d, md)$ from the sidelobes of the neighboring correlations. To overcome such large SBP requirements, a holographic approach using a volume material of R^3 resolvable points capable of satisfying the N^4 requirements where $N^4 = R^3$ has been suggested [6]. In this paper we propose to employ random phase coded optical correlation [7], which will be discussed in the following sections.

3. Matrix-tensor Multiplication Based on Random Phase Coding

An optical correlator with the same random phase code incorporated into the two functions to be correlated provides at the origin of the output the inner product value, while the energy associated with the sidelobes of the correlation function is scattered over the entire output plane. To use this concept in implementing a matrix tensor multiplier, one can employ the random phase code and the shift theorem as described in the following. The correlation function will be given by

$$c(x,y) = g(x,y)e^{j\phi(x,y)} \circledast h(x,y), \tag{3}$$

where $\phi(x,y)$ is a random phase function and

$$h(x,y) = \sum_{\ell=1}^{L} \sum_{m=1}^{M} W_{\ell m}(x-\ell\Delta, y-m\Delta) e^{j\phi(x-\ell\Delta, y-m\Delta)}. \tag{4a}$$

Here Δ is the sampling period of c(x,y) and the shift interval of the tensor components, and is defined by

$$\Delta \triangleq K\delta; \tag{4b}$$

Here δ is the sampling period of the random phase process and K is the number of random phase samples in interval Δ. The tensor encoding according to Eq. (4) is shown in Fig. 2a.

To show that the correlation function given by Eq. (3) provides the desired output of the matrix-tensor multiplier, we will determine its expectation values (see also refs. 7-9):

$$E\{c(x, y)\} = \sum_{\ell=1}^{L} \sum_{m=1}^{M} \left[\iint_{-\infty}^{\infty} g(\xi-\ell\Delta, \eta-m\Delta) W_{\ell m}(\xi-\ell\Delta, \eta-m\Delta) d\xi d\eta \right] \cdot P(x-\ell\Delta, y-m\Delta)$$

$$= \sum_{\ell=1}^{L} \sum_{m=1}^{M} C_{\ell m} P(x-\ell\Delta, y-m\Delta), \tag{5a}$$

where

$$C_{\ell m} = \iint_{-\infty}^{\infty} g(\xi-\ell\Delta, \eta-m\Delta) W_{\ell m}(\xi-\ell\Delta, \eta-m\Delta) d\xi d\eta \tag{5b}$$

is the inner product of the input function and the ℓm-th component of the tensor $\underline{W}(x,y)$, E denotes the expectation value, $P(x,y) = |a(x,y)|^2$ and a(x,y) is the impulse response of the optical system which is usually a real function. The function P(x,y) describes the blurring effect due to the finite bandwidth of the channel. For an ideal optical system P(x,y) is a delta function. Equation (5a) can be interpreted as a set of LM pulses spaced at a distance Δ, while the amplitude of each pulse is equal to the inner product given by Eq. (5b) (see Fig. 2b). To satisfy the necessary condition to separate and to resolve the inner products, we will require the spatial width of the function P(x,y) to be narrower than δ.

However, due to the fact that the random phase coding process is used to reduce the SBP requirements we are also introducing a noise term to the output, which will reduce the output *S/N* ratio. The intensity (*S/N*) ratio can be defined as

$$(S/N)_I \triangleq \left\{ \frac{E\{c(x,y)\}}{(\mathrm{Var}\ \{c(x,y)\})^{1/2}} \right\}^2 . \tag{6a}$$

The *S/N* ratio of Eq. (6a) can be employed in evaluating the dynamic range of the system, which is defined as the output ratio of the maximum value of the signal to the maximum value of the noise. The resultant dynamic range D (see Appendix and refs. 7-9) is given by

$$D = \frac{4K^2 SBP_M}{LM} . \tag{6b}$$

The relation given in Eq. (6b) has been obtained under the assumption that the functions g(x,y) and $W_{fm}(x,y)$ are real and normalized.

Applying the matrix-tensor multiplier to perform the functions of a fully interconnected network ($L=M=N$, $SBP_M = N^2$)

$$D = 4K^2 . \tag{6c}$$

The phase coded matrix-tensor multiplier will require an optical channel SBP given by

$$SBP_{MT}' = K^2[\sqrt{SBP_M} + (L-1)][\sqrt{SBP_M} + (M-1)] \tag{7a}$$

Again for our example of a fully interconnected network ($L=M=N$, $SBP_M \cong N^2$) we obtain

$$SBP_{MT}' \cong 4K^2 SBP_M = 4K^2 N^2 . \tag{7b}$$

Comparing Eq. (2a) with (7a), one can conclude that the SBP requirements of the channel using a phase-coded matrix-tensor multiplier are smaller than the ones using pure space multiplexing. For the example of a fully interconnected network the reduction in the channel SBP is given by

$$\frac{SBP_{MT}}{SBP_{MT}'} = \frac{N^2}{4K^2} . \tag{8}$$

4. SBP-Dynamic Range Trade-offs

The matrix-tensor multiplier based on space multiplexing requires a large SBP (Eq. 2), but offers very large dynamic range at the output (e.g., $D > N^2$). The matrix-tensor multiplier based on random phase coding will require less SBP (Eq. 7) at the expense of the dynamic range (Eq. 6). Therefore, there exists a trade-off between the available SBP of the channel, and the dynamic range required by a specific optical system. To illustrate these trade-offs we will give two examples, where the limiting elements of the SBP of the optical channel are the nonlinear materials in which the correlations are performed. The two types of nonlinear materials are the planar (e.g., 2-D holographic materials) and the volume (e.g., 3-D photorefractive crystals or volume holographic materials). We will assume that the planar and the volume materials will be capable of supporting an optical channel with a SBP of R^2 and R^3 respectively. In our examples we will also assume a fully interconnected network, $L=M=\sqrt{SBP_M}=N$, i.e., the N^2 inputs are connected to the N^2 outputs via the N^4 connections of the fourth rank tensor. For a planar material, the maximum size of the input/output arrays for conventional, (N_c) and phase coded, (N_p) matrix-tensor multiplication will be given by

$$N_c = R^{1/2} \tag{9a}$$

and

$$N_p = \frac{R}{2K} = \frac{R}{2\sqrt{D}} \tag{9b}$$

respectively. The plots comparing the two result of Eqs. (9) are shown in Fig. 3 in a log-log scale. Similarly, for a volume material we obtain

$$N_c = R^{3/4} \tag{10a}$$

$$N_p = \frac{R^{3/2}}{2\sqrt{D}} \tag{10b}$$

The comparison plots for a volume material are shown in Fig. 4. One can clearly observe from Fig. 3 and Fig. 4 that there is a trade-off between the dynamic range of the multiplier and the size of the array at which the system can be reliably operated. For example, for a D = 100 and planar material we will be better off using the phase coded technique for input/output arrays larger than (20× 20) pixels. Another way of observing the trade-off is to note that in order to operate on an array of 1000 × 1000 pixels at D = 100 using phase coding, we will need a planar material of SBP of only $2{\cdot}10^4 \times 2{\cdot}10^4$ pixels as opposed to $10^6 \times 10^6$ pixels using the conventional technique. Furthermore, a volume material will enhance even more the advantages of the phase coded technique over the conventional one. For example, to operate on an array of 1000 × 1000 pixels at D = 100 a volume material will require SBP of $10^3 \times 10^3 \times 10^3$ for the phase coded technique and $10^4 \times 10^4 \times 10^4$ for the conventional technique, respectively.

5. Conclusions

We have shown that the random phase coded technique is very useful in computing matrix-tensor operations because it allows us to operate on larger matrix arrays for a given SBP of nonlinear material in numerous optical signal processing applications. The analysis of the trade-off between the available SBP and the output dynamic range shows that it is possible to construct systems with available real time holgraphic materials (e.g., photorefractive materials, organic materials, etc.) for multiplication of large 2-D arrays of data with a fourth rank tensor in real time. Systems, for which the dynamic range of D = 100 is quite sufficient, are of high importance in the areas of neural optical computing and reconfigurable optical interconnects for digital optical processing applications.

Acknowledgment

The authors greatly acknowledge the research support of DARPA, administered by the AFOSR under contract No. 88-0022, ONR under contract No. N00014-86-K-0697 and AFOSR under contract No. 85-0371, and valuable discussions with D. Psaltis, J. Hong of California Institute of Technology and J. Jau of UCSD.

Appendix

In this appendix we will study the phase coded technique by analyzing the S/N ratio at the output defined by Eq. (6a). For simplicity we will conduct the analysis for a one-dimensional case. The S/N ratio of the amplitude of c(x) is defined by

$$(S/N)_A = \frac{E\{c(x)\}}{(\mathrm{Var}\ \{c(x)\})^{1/2}} = \frac{E\{c(x)\}}{(E\{(c(x)-E\{c(x)\})^2\})^{1/2}} \qquad (A.1)$$

where E denotes the expectation value, $E\{c(x)\}$ is the first moment and the variance is determined from

$$\mathrm{Var}\ \{c(x)\} = E\{(c(x)-E\{c(x)\})^2\}$$
$$= E\{c^2(x)\} - (E\{c(x)\})^2 \qquad (A.2)$$

To estimate the (S/N) ratio we therefore have to determine the first and the second moments of the correlation function c(x).

First moment

The correlation function of eqs. (3) and (4) can be rewritten in an alternative form

$$c(x) = g(x)\,e^{j\phi(x)} * \sum_{\ell=1}^{L} W_\ell^*(x-\ell\Delta)\,e^{-j\phi(x-\ell\Delta)} \circledast P(x)$$

$$= g(x)\,e^{j\phi(x)} * \sum_{\ell=1}^{L} \left\{ \int_{-\infty}^{\infty} \delta(y-\ell\Delta)\,W_\ell^*(x-y)\,e^{-j\phi(x-y)}\,dy \right\} \circledast P(x) \qquad (A.3)$$

where $P(x) = |a(x)|^2$ and a(x) is the impulse response of the optical system and is a real function. Calculating the first moment,

$$E\{c(x)\} = E\left\{ \left[\sum_{\ell=1}^{L} \int_{-\infty}^{\infty} dy \int_{-\infty}^{\infty} ds\ \delta(y-\ell\Delta)\,W_\ell^*(s-y)\,e^{-j\phi(s-y)}\,g(s-x)\,e^{j\phi(s-x)} \right] \circledast P(x) \right\} \qquad (A.4)$$

Under the assumption that the phase $\phi(x)$ is a random variable uniformly distributed in the interval $[-\pi, \pi]$, the last equation can be rewritten as

$$E\{c(x)\} = \sum_{\ell=1}^{L} \int_{-\infty}^{\infty} dy \int_{-\infty}^{\infty} ds \int_{-\infty}^{\infty} dz\, \delta(y-\ell\Delta)\, W_\ell^*(x-y)\, g(s-z)$$

$$\delta(z-y)\, P(z-x) \tag{A.5}$$

where we employed the relation [10]

$$E\{e^{-j\phi(s-y)}\, e^{j\phi(s-x)}\} = \delta(x-y) \tag{A.6}$$

Finally the equation (A.5) can be written as

$$E\{c(x)\} = \sum_{\ell=1}^{L} P(x-\ell\Delta) \left\{ \int_{-\infty}^{\infty} W_\ell^*(s-\ell\Delta)\, g(s-\ell\Delta)\, ds \right\} \tag{A.7}$$

The last relation can be interpreted as a set of L pulses spaced at a distance Δ, while the amplitude of each pulse equals to the inner product between the input signal s(x) and the ℓ-th component of $\underset{\sim}{W}(x)$. The term $P(x-\ell\Delta)$ describes the blurring effect due to the finite channel bandwidth (ideally P(x) would be a δ-function if the channel bandwidth were infinite). To separate spatially the inner products we have to satisfy the condition that the spatial extension (width) of the function P(x) is much smaller than the sampling distance, Δ.

Second moment

The second moment is determined from the correlation function c(x)

$$E\{c^2(x)\} = E\{c(x_1)\, c^*(x_2)\}\big|_{x_1=x_2=x} \tag{A.8}$$

The correlation function c(x) is given by

$$E\{c(x_1)c^*(x_2)\} = E\left\{\sum_{\ell_1=1}^{L}\sum_{\ell_2=1}^{L}\iint_{-\infty}^{\infty} dy_1 ds_1\ \delta(y_1-\ell_1\Delta)\, W_{\ell_1}{}^*(s_1-y_1)\, g(s_1-x_1)\cdot\right.$$

$$\cdot\ e^{-j\phi(s_1-y_1)}\, e^{j\phi(s_1-x_1)} \iint_{-\infty}^{\infty} dy_2 ds_2\ \delta(y_2-\ell_2\Delta)\cdot$$

$$\left.\cdot\ W_{\ell_2}(s_2-y_2)\, g^*(s_2-x_2)\, e^{j\phi(s_2-y_2)}\, e^{-j\phi(s_2-x_2)}\right\} \circledast P(x_1)\,P^*(x_2). \qquad \text{(A.9)}$$

Using the relation

$$E\,\{e^{-j\phi(s_1-y_1)}\, e^{j\phi(s_1-x_1)}\, e^{j\phi(s_2-y_2)}\, e^{-j\phi(s_2-y_2)}\}$$

$$= \delta(y_1-x_1)\,\delta(x_2-y_2) + \delta(s_2-s_1+y_1-y_2)\,\delta(s_1-s_2+x_2-x_1) \qquad \text{(A.10)}$$

Eq. (A.9) can be rewritten as

$$E\{c(x_1)c(x_2)\} = J_1 + J_2 \qquad \text{(A.11)}$$

where

$$J_1 = \{\ \cdots\ \delta(y_1-x_1)\,\delta(x_2-y_2)\ \cdots\ \} \circledast P(x_1)\,P^*(x_2)$$

and

$$J_2 = \{\ \cdots\ \delta(s_2-s_1+y_1-y_2)\,\delta(s_1-s_2+x_2-x_1)\ \cdots\ \} \circledast P(x_1)\,P^*(x_2)$$

Evaluating the integrals in terms J_1 and J_2 we obtain:

$$J_1 = \sum_{\ell_1=1}^{L} \sum_{\ell_2=1} P(x_1 - \ell_1\Delta) P^*(x_2 - \ell_2\Delta) \int_{-\infty}^{\infty} ds_1\, W_{\ell_1}^*(s_1 - \ell_1\Delta) \cdot$$

$$\cdot\, g(s_1 - \ell_1\Delta) \int_{-\infty}^{\infty} ds_2\, W_{\ell_2}(s_2 - \ell_2\Delta)\, g^*(s_2 - \ell_2\Delta) \tag{A.12a}$$

$$J_2 = \int_{-\infty}^{\infty} dZ_1 \sum_{\ell_1=1}^{L} \sum_{\ell_2=1}^{L} \int_{-\infty}^{\infty} ds_1\, W_{\ell_1}{}^*(s_1 - \ell_1\Delta)\, g(s_1 - z_1)\, W_{\ell_2}(s_1 - \ell_1\Delta)$$

$$\cdot\, g^*(s_1 - z_1)\, P(z_1 - x_1)\, P^*(z_1 - (\ell_2 - \ell_1)\Delta - x_2) \tag{A.12b}$$

Substituting $x_1 = x_2 = x$ the last equations can be rewritten as

$$J_1 = \sum_{\ell_1=1}^{L} \sum_{\ell_2=1}^{L} P(x - \ell_1\Delta) P^*(x - \ell_2\Delta) \int_{-\infty}^{\infty} ds_1\, W_{\ell_1}{}^*(s_1 - \ell_1\Delta)$$

$$g(s_1 - \ell_1\Delta) \int_{-\infty}^{\infty} ds_2\, W_{\ell_2}(s_2 - \ell_2\Delta)\, g^*(s_2 - \ell_2\Delta). \tag{A.13a}$$

$$J_2 = \sum_{\ell_1=1}^{L} \sum_{\ell_2=1}^{L} \int_{-\infty}^{\infty} ds_1 \cdot W_{\ell_1}^*(s_1 - \ell_1\Delta)\, W_{\ell_2}(s_1 - \ell_1\Delta)$$

$$\int_{-\infty}^{\infty} dz_1\, g(s_1 - z_1)\, g^*(s_1 - z_1)\, P(z_1 - x)\, P^*(z_1 - x + (\ell_2 - \ell_1)\Delta) \tag{A.13b}$$

Assuming P(x) to be a narrow function we obtain:

$$P(x - \ell'\Delta) P^*(x - \ell_2\Delta) = \begin{cases} |P(x - \ell'\Delta)|^2 & \ell_1 = \ell_2 = \ell' \\ 0 & \ell_1 \neq \ell_2 \end{cases} \tag{A.13c}$$

and

$$P(z_1 - x) P^*(z_1 - x + (\ell_2 - \ell_1)\Delta) = \begin{cases} |P(z_1 - x)|^2 & \ell_1 = \ell_2 = \ell' \\ 0 & \ell_1 \neq \ell_2 \end{cases} \tag{A.13d}$$

Substituting these last relations into J_1 and J_2 we obtain

$$J_1 = \sum_{\ell'=1}^{L} |P(x-\ell'\Delta)|^2 \left| \int_{-\infty}^{\infty} W_{\ell}^{*}(s-\ell'\Delta)\, g(s-\ell'\Delta)\, ds \right|^2 \tag{A.14a}$$

$$J_2 = \sum_{\ell'=1}^{L} \iint dz_1 ds_1 \, |W_{\ell'}(s_1-\ell'\Delta)|^2 \, |g(s_1-z_1)|^2 \, |P(z_1-x)|^2 \tag{A.14b}$$

To evaluate the variance given by Eq. (A.2) we first calculate the second term $E\{c(x)\})^2$ using the result of Eq. (A.7),

$$|E\{c(x)\}|^2 = \left| \sum_{\ell=1}^{L} P(x-\ell\Delta) \left\{ \int_{-\infty}^{\infty} W_{\ell}(s-\ell\Delta)\, g(s-\ell\Delta)\, ds \right\} \right|^2$$

$$= \sum_{\ell_1=1}^{L} \sum_{\ell_2=1}^{L} P(x-\ell_1\Delta)\, P^*(x-\ell_2\Delta) \left\{ \int_{-\infty}^{\infty} W_{\ell_1}^{*}(s-\ell_1\Delta)\, g(s-\ell_1\Delta)\, ds \right\}$$

$$\cdot \left\{ \int_{-\infty}^{\infty} W_{\ell_2}(s-\ell_2\Delta)\, g^*(s-\ell_2\Delta)\, ds \right\}$$

$$\cong |P(x-\ell'\Delta|^2 \left| \int_{-\infty}^{\infty} W_{\ell}(s-\ell'\Delta)\, g(s-\ell'\Delta)\, ds \right|^2 \tag{A.15}$$

where we have used the result of Eq. (A.13c) with P(x) being a narrow function. Comparing (A.15) with J_1 from Eq. (A.14a) the variance from Eq (A.2) can be determined as

$$\mathrm{Var}\ \{c\} = J_2. \tag{A.16}$$

Considering again that the function $P(z-x)$ is much narrower than $g(x)$, we evaluate the term J_2 as

$$\text{Var}\ \{c(x)\} = \sum_{\ell'=1}^{L} \int_{-\infty}^{\infty} ds_1 \,|W_{\ell'}(s_1 - \ell'\Delta)|^2$$

$$\int_{-\infty}^{\infty} dz_1 \,|g(s_1 - z_1)|^2 \,|P(z_1 - x)|^2$$

$$= \sum_{\ell'=1}^{L} \int_{-\infty}^{\infty} ds_1 \,|W_{\ell'}(s_1 - \ell'\Delta)|^2 \,|g(s_1 - x)|^2 \int_{-v/2}^{v/2} |P(\xi)|^2 \, d\xi \qquad \text{(A.17)}$$

where v is the width of P(x).

We are interested in evaluating the S/N ratio at a point $x = \ell\Delta$, and therefore we evaluate the variance of c(x) at these locations,

$$\text{Var}\ \{c(x{=}\ell\Delta)\} = \sum_{\ell'=1}^{L} \int_{-\infty}^{\infty} ds \,|W_{\ell'}(s_1 - \ell'\Delta)|^2 \,|g(s_1 - \ell\Delta)|^2$$

$$\cdot \int_{-v/2}^{v/2} |P(\eta)|^2 d\eta \qquad \text{(A.18)}$$

The S/N ratio can be employed in evaluating the dynamic range of the system, which is defined as the output ratio of the maximum value of the signal to the maximum value of the noise. The maximum value of the first moment at a location $x = \ell\Delta$, will be determined by

$$E\{c(x)\} = X \qquad \text{(A.19)}$$

where X is the spatial extension of the input information.

The maximum value of the variance at the origin is given by

$$\text{Var}\ \{c(0)\} = v \sum_{\ell'=1}^{L} \frac{X}{L}\,\ell' = v\,\frac{X}{2}\,(L+1) \qquad \text{(A.20)}$$

or at any $x = \ell\Delta$

$$\text{Var}\ \{c(x{=}\ell\Delta)\} = vX \sum_{\ell'-\ell=1}^{L-\ell} \frac{\ell'-\ell}{L} \qquad \text{(A.21)}$$

The variance reaches its maximum at the origin, x = 0, and therefore the amplitude dynamic range, D_A will satisfy the following inequality

$$D_A \geq \frac{X}{\sqrt{\delta \frac{X}{2} (L+1)}} \tag{A.22}$$

where we have assumed that the sampling period of the phase process, δ is equal to the width v of the function P(x). Considering the relations

$$\Delta = \delta K$$

and

$$SBP_M = \frac{X}{\Delta}$$

the dynamic range of eq. (A.21) can be rewritten as

$$D_A \geq \sqrt{\frac{2\, SBP_M \cdot K}{L}}$$

Furthermore, the dynamic range of the intensity D, for a 2-D case will be given by

$$D \geq \frac{4\, SBP_M \cdot K^2}{LM} \tag{A.23}$$

References

1. J. W. Goodman, A. R. Dias, and L. M. Woody, "Fully parallel, high-speed incoherent optical method for performing discrete Fourier transform", *Opt. Lett.*, 2, 1-3, 1978.
2. D. Psaltis, D. Casasent, and M. Carlotto, "Iterative color-multiplexed, electro-optical processor", *Opt. Lett.*, 4, 348-3501, 1979.
3. H. Rajbenbach, Y. Fainman, and Sing H. Lee, "Optical implementation of an iterative algorithm for matrix inversion", *Appl. Opt.*, 26, 1024-1031, 1987.
4. J. W. Goodman, F. J. Leonberger, S. Y. Kung, and R. A. Athale, "Optical interconnects for VLSI", *Proc. IEEE*, 72, 850-866, 1984.
5. D. Psaltis and N. Farhat, "Optical information processing based on an associative-memory model of neural nets with thresholding and feedback", *Opt. Lett.*, **10**, 98-100, 1985.
6. J. Hong and D. Psaltis, "Storage capacity of holographic associative memories", *Opt. Lett.* **11**, 812-814, 1986. See also: D. Psaltis, J. Yu, X. G. Gu, and H. Lee, "Optical neural nets implemented with volume holograms", Topical meeting on Optical Computing, March 1987, *Technical Digest*, **11**, 129-132, 1987.
7. J. R. Leger and Sing H. Lee, "Hybrid optical processor for pattern recognition and classification using generalized set of pattern functions", *Appl. Opt.* **21**, 274-287, 1982. (See Eq. 7.)
8. Y. Fainman, J. Shamir, and E. Lenz, "Static and dynamic behavior of speckle patterns described by operator algebra", *Appl. Opt.* **20**, 3516-3538, 1981. (See Sections IVA, D and V, Eq. 71.)
9. E. L. Kral, J. F. Walkup, and M. O. Hagler, "Correlation properties of random phase diffusers for multiplex holography", *Appl. Opt.*, **21**, 1281-1290, 1982. (See Eqs. 51 and 52.)
10. J. Shamir and Y. Fainman, "Speckle pattern correlation with double-beam and polychromatic illumination," *Appl. Opt.*, **23**, 1547-15531, 1984.

Figure Captions

Fig. 1. (a) Representation of the tensor by space multiplexing. The components of the tensor are space-multiplexed with a period d, where d is equal to or larger than the spatial extension ($N\Delta$) of the tensor's components, $W_{\ell m}(x,y)$.

Fig. 1. (b) The output of matrix-tensor multiplication based on the space multiplexing techniques.

Fig. 2. (a) Representation of the tensor by random phase coding. The components of the tensor $W_{\ell m}(x,y)$ of spatial extension ($N\Delta$) are multiplied by a random phase code which has a pixel resolution of δ. The tensor components are shifted with respect to each other with a period $\Delta = K\delta$ and then superimposed on one another.

Fig. 2. (b) The output of matrix-tensor multiplication based on the random phase coding technique.

Fig. 3. Comparison of phase coded and conventional matrix-tensor multiplication using planar material. R^2 is the available space bandwidth product (SBP) provided by the real-time nonlinear material (RTNM); N^2 is the size of the input/output 2-D array. To operate on a matrix of 1000×1000 the conventional method will require a SBP of $10^6 \times 10^6$ from the RTNM, while the phase coded M-T multiplier will require only a SBP of $2\cdot 10^4 \times 2\cdot 10^4$ for D = 100 and $6\cdot 10^3 \times 6\cdot 10^3$ for D = 10 from RTNM. Alternatively, given a SBP of 1000×1000 for the RTNM, the conventional M-T multiplier can accomodate a matrix of 32x32, while the phase coded M-T multiplier will be able to accomodate a *50×50 matrix for D = 100 and 160×160 matrix for D =* 10.

Fig. 4. Comparison of phase coded and conventional matrix-tensor multiplication using volume material. R^3 is the available space bandwidth product (SBP) provided by real-time nonlinear material (RTNM); N^2 is the size of the input/output 2-D array. To operate on a matrix of 1000×1000 arrays a conventional method will require a SBP of $10^4 \times 10^4 \times 10^4$ from RTNM, while the phase coded multiplier will require a SBP less than $10^3 \times 10^3 \times 10^3$ for D = 100 and a SBP of less than 400×400×400 for D = 10 from RTNM.

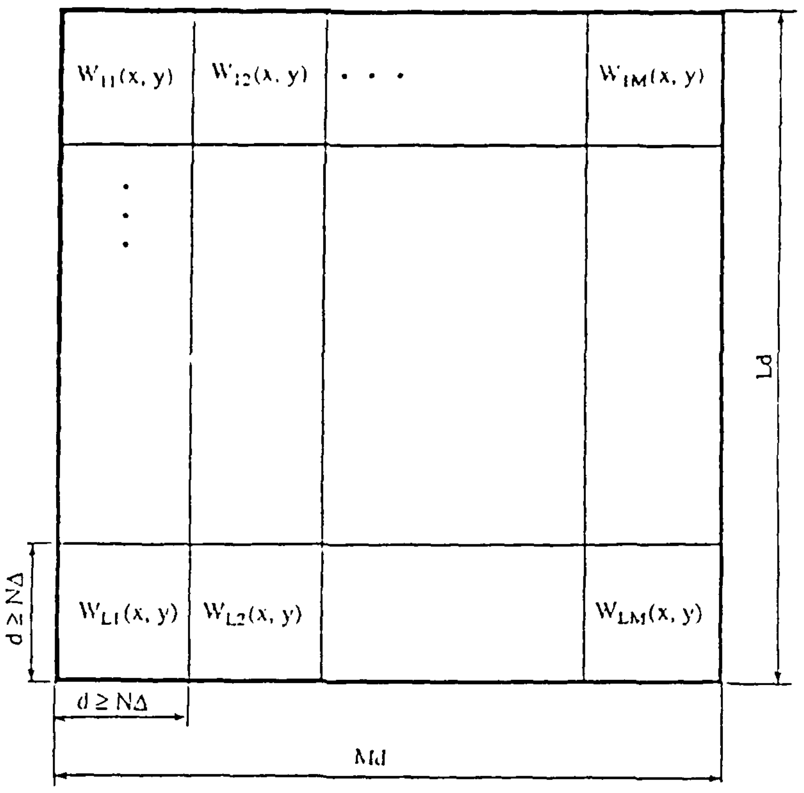


Fig. 1. (a) Representation of the tensor by space multiplexing. The components of the tensor are space-multiplexed with a period d, where d is equal to or larger than the spatial extension ($N\Delta$) of the tensor's components, $W_{\ell m}(x,y)$.

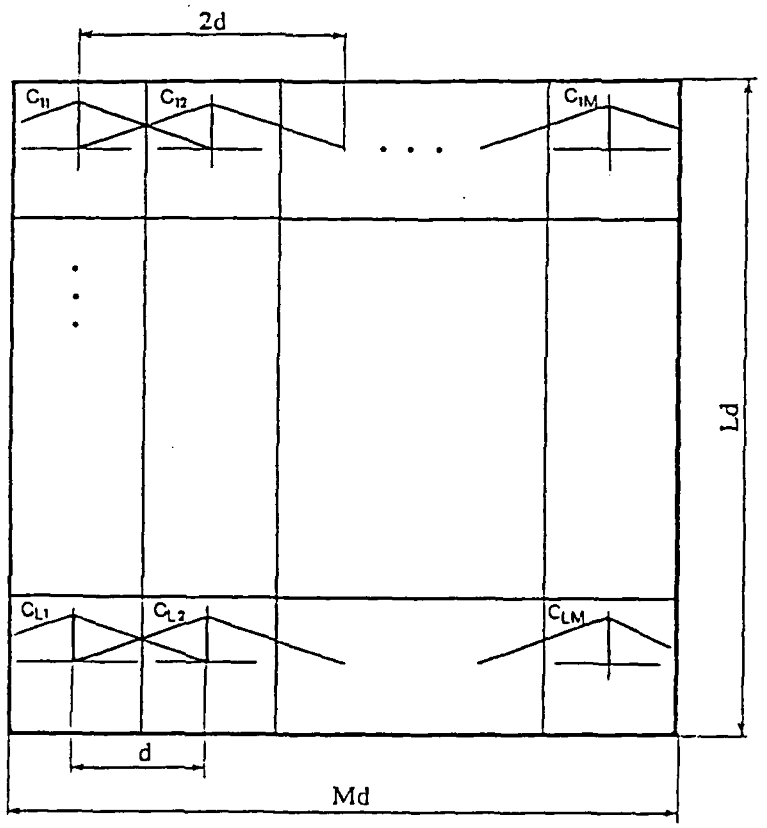


Fig 1. (b) The output of matrix-tensor multiplication based on the space multiplexing techniques.

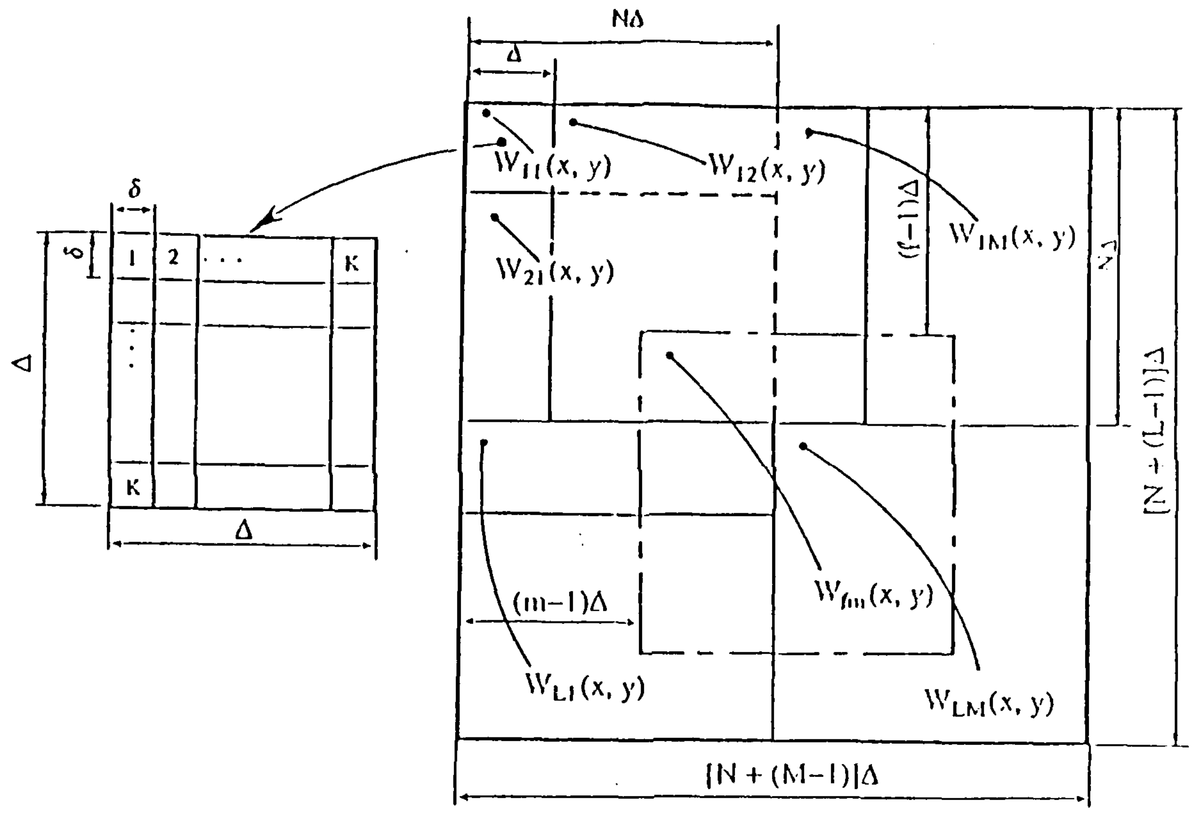


Fig. 2. (a) Representation of the tensor by random phase coding. The components of the tensor $W_{fm}(x,y)$ of spatial extension $N\Delta$ are multiplied by a random phase code which has a pixel resolution of δ. The tensor components are shifted with respect to each other with a period $\Delta = K\delta$ and then superimposed on one another.

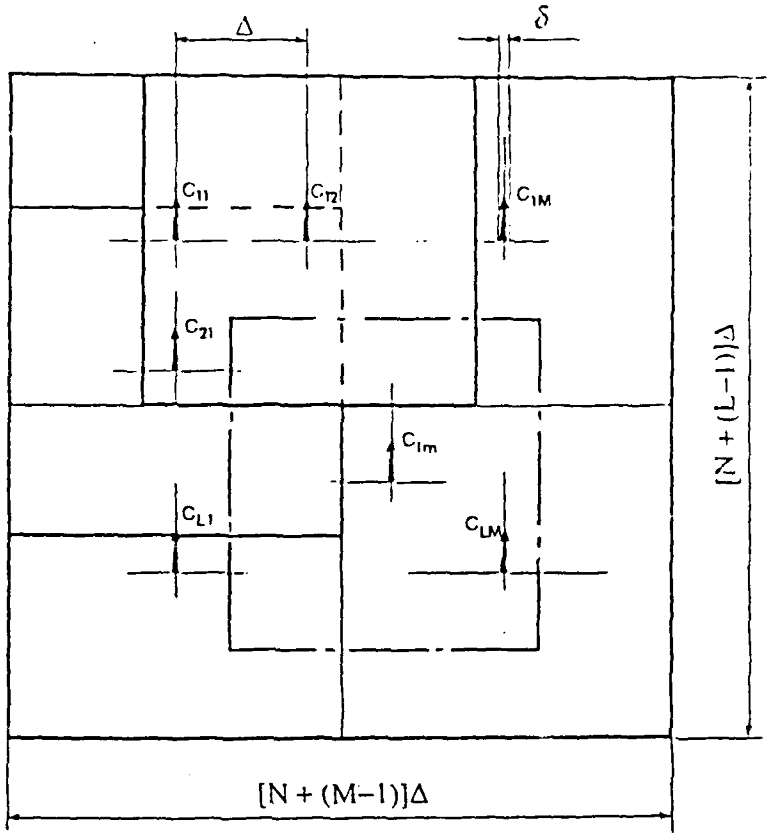


Fig. 2. (b) The output of matrix-tensor multiplication based on the random phase coding technique.

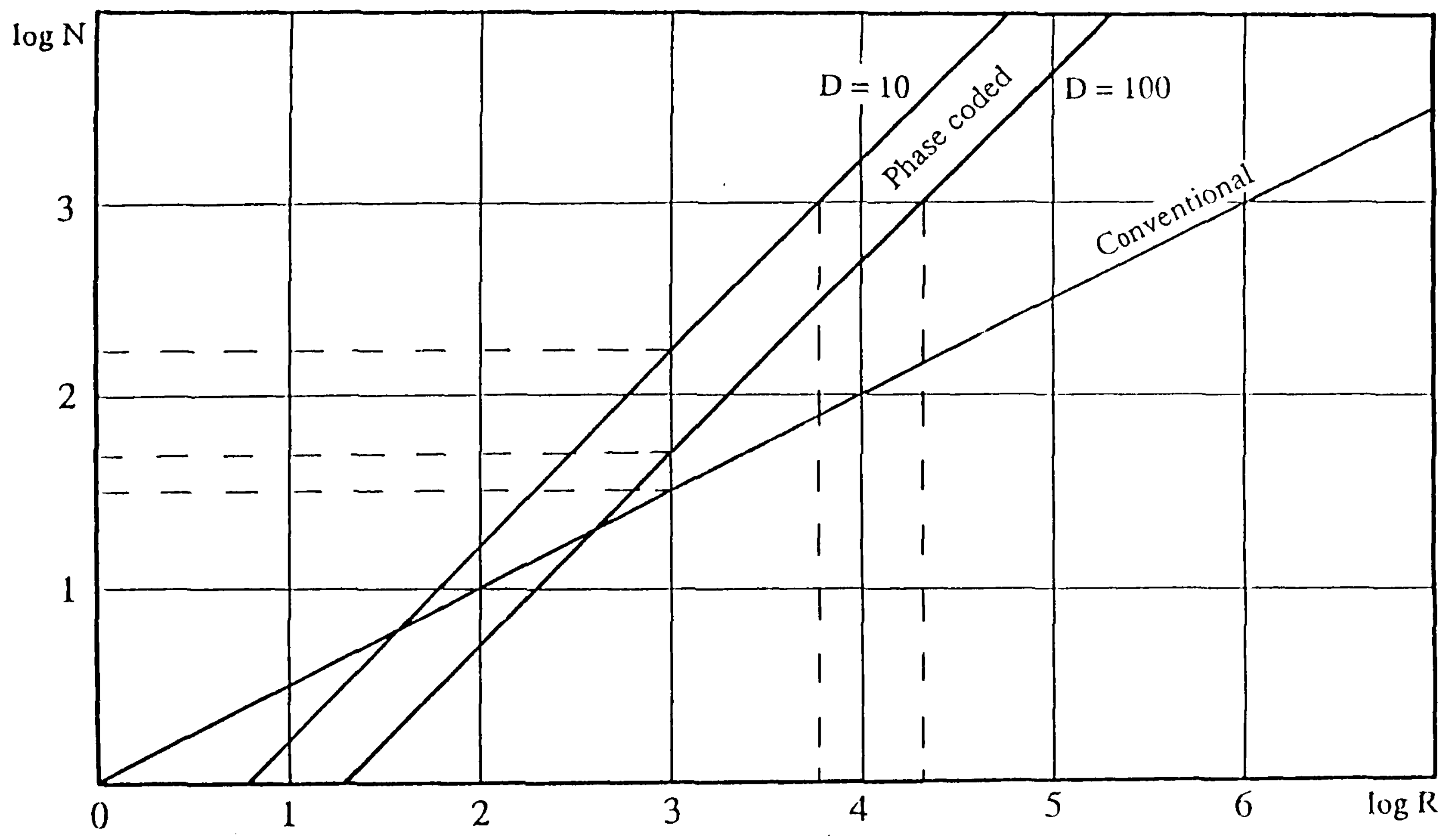


Fig. 3. Comparison of phase coded and conventional matrix-tensor multiplication using planar material. R^2 is the available space bandwidth product (SBP) provided by the real-time nonlinear material (RTNM); N^2 is the size of the input/output 2-D array. To operate on a matrix of *1000×1000 the conventional method will require a SBP of* $10^6 \times 10^6$ *from the* RTNM, while the phase coded M-T multiplier will require only a SBP of $2{\cdot}10^4 \times 2{\cdot}10^4$ for D = 100 and $6{\cdot}10^3 \times 6{\cdot}10^3$ for D = 10 from RTNM. Alternatively, given a SBP of 1000×1000 for the RTNM, the conventional M-T multiplier can accomodate a matrix of 32×32, while the phase coded M-T multiplier will be able to accomodate a 50×50 matrix for D = 100 and 160×160 matrix for D = 10.

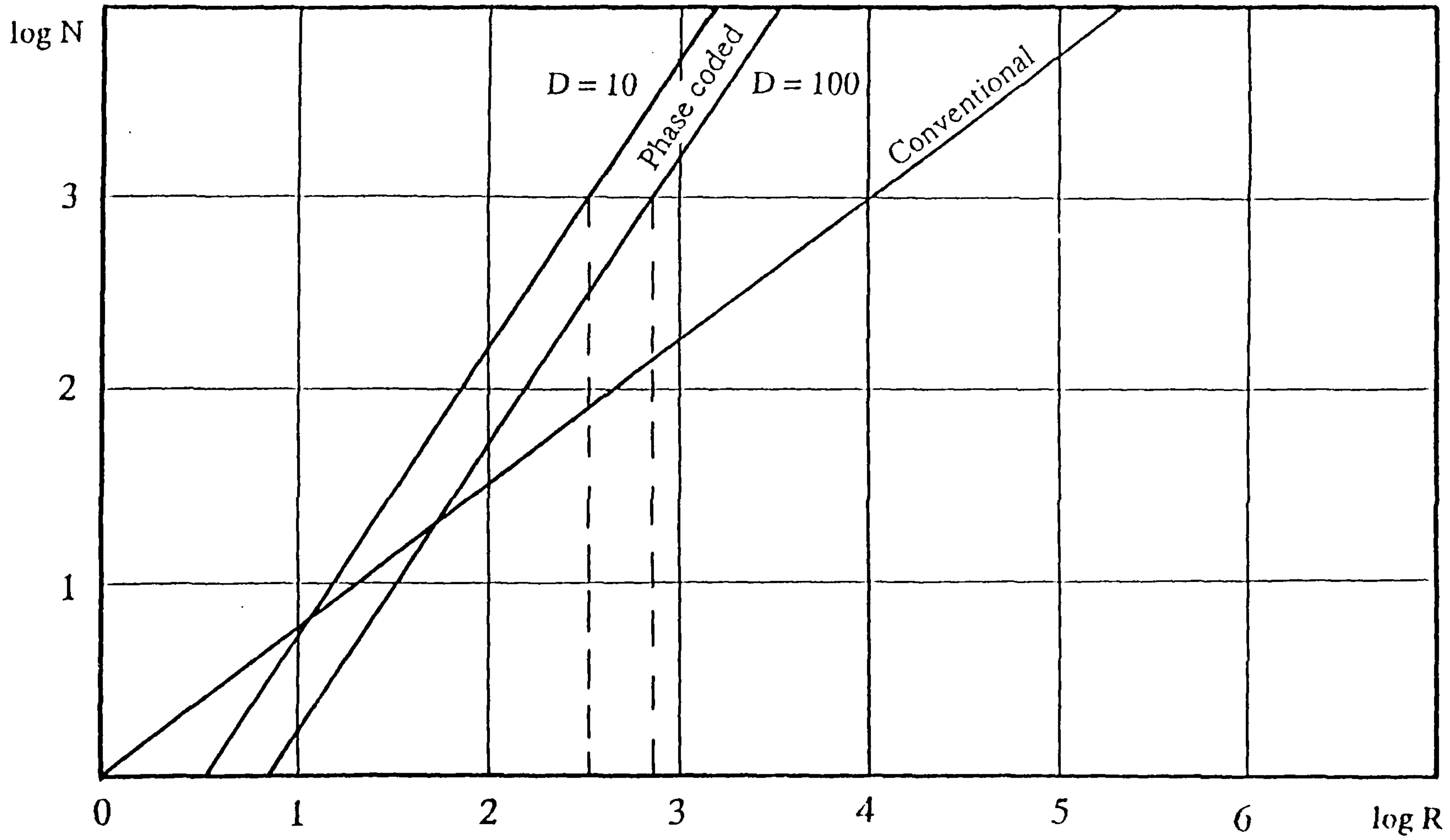


Fig. 4. Comparison of phase coded and conventional matrix-tensor multiplication using volume material. R^3 is the available space bandwidth product (SBP) provided by real-time nonlinear material (RTNM); N^2 is the size of the input/output 2-D array. To operate on a matrix of 1000×1000 arrays a conventional method will require a SBP of $10^4 \times 10^4 \times 10^4$ from RTNM, while the phase coded multiplier will require a SBP less than $10^3 \times 10^3 \times 10^3$ for D = 100 and a SBP of less than 400×400×400 for D = 10 from RTNM.

END
DATED
FILM
8-88
DTIC